Joint source-channel turbo decoding of VLC encoded Markov sources

Arnaud Guyader, Eric Fabre, Christine Guillemot

Irisa-Inria
Campus universitaire de Beaulieu, 35042 Rennes Cedex, France
firstname.lastname@irisa.fr

Résumé – Ce papier décrit un algorithme de décodage conjoint pour une chaîne de transmission composée d'une source markovienne, d'un codeur de source et d'un codeur de canal. Le modèle global de la chaîne n'est pas directement manipulable, aussi s'oriente-t-on vers une procédure itérative traitant séparément chaque composant. On s'intéresse notamment aux difficultés induites par les mots de code de longueur variable, et l'on montre qu'un décodage séquentiel des deux premiers composants est optimal.

Abstract – We propose a joint decoding procedure for a chain composed of a Markov source, a source coder and a channel coder. Global processing of the chain is intractable, hence we design an iterative procedure alternating use of each component. Special attention is devoted to the variable length source coder ; we show that sequential soft decoding for the first two components is optimal.

1 Why joint decoding ?

Joint source and channel coding has gained considerable attention as a viable alternative to the traditional separated approach for reliable communication over noisy channels. Joint coding is particularly suited to heavily constrained transmission systems in terms of complexity, frame length, decoding delay, etc. This idea often relies on capitalizing on the source coder suboptimality (the so-called "excess-rate") to reduce the complexity of the channel coder. As a consequence, the receiver must decode the observed noisy bitstream by using jointly the redundancy introduced by the channel code and the residual redundancy of the source.

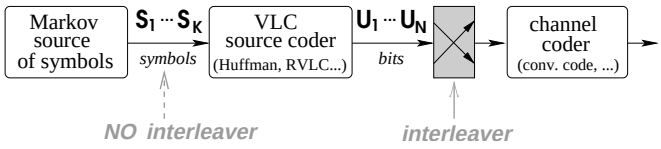

Fig. 1: *Coding chain.*

The wide usage of variable length codes (VLCs) for source coding (i.e. data compression), which are very sensitive to channel noise, has already motivated several procedures for robust joint decoding of variable length coded streams [1, 2, 3, 4, 5, 6]. In this paper, we also address the joint decoding problem along these lines, assuming the general transmission chain of fig. 1. This chain is composed of 1/ a Markov source, 2/ a variable length source coder and 3/ a convolutional channel coder. We focus on the analysis and modeling of the dependencies between the variables involved in the complete source and channel coding chain, by means of the Bayesian-network formalism.

Our starting point is a state space model of the three different components. These models are cascaded to produce the bitstream sent over the channel, and the randomness of variables is introduced by assuming a white noise input of the cascade. The product of these three automata induces immediately a state variable model of the bitstream : the triple of states – one state for each model – appears to be a Markov chain, the transitions of which generate the sequence of output bits, that are sent over the channel. The observed output of a memoryless channel corresponds to noisy measurements of these bits, so we are exactly in the usual HMM framework.

This nice picture suffers from two difficulties however. First, the presence of two time indexes : the symbol clock of the source model, and the bit clock of the channel coder model. The translation is performed by the VLC source coder. Since not all symbols have the same length, the number of bits of the coded sequence (as well as the position of symbol boundaries) is a random variable, which is quite unusual. We therefore have to solve a joint segmentation + estimation problem. The second difficulty is more classical : it comes from the fact that the state space dimension of the product model explodes in most practical cases, so that a direct application of usual estimation algorithms for HMMs is unaffordable, except in trivial cases [3].

In this paper, we thus rely on properties evidenced by serial turbo-codes to design an estimation strategy : instead of using the big product model, inference can be done in an iterative way, making use of part of the global model at each time. This decreases complexity since smaller state spaces are involved. We use this idea in the following way, as it was already suggested in [5] and [6] : we introduce an interleaver between the source coder and the

channel coder. This allows the construction of an iterative soft decoder alternating between the channel coder model and the joint model of the source + source coder [1], with the bit clock as time index. But the idea can be pushed further: why not splitting also the joint model source + source coder? We demonstrate that this can be done without invoking a turbo procedure. Due to the pointwise translation of symbols into bits, there is no need of an interleaver there. The soft decoding for this pair of models can be performed *optimally* by using first the source coder model, assuming the symbol source is white, and then incorporating the source model to take symbol correlations into account. This property is obvious for constant length source codes, but it is very surprising that it still holds for VLCs. We end up with a turbo-like decoding algorithm alternating the use of the two sources of redundancy in the bitstream: the Markov source on the one hand, the channel coder on the other hand. The intermediary VLC source coder model is used as a translator of soft information from the bit clock to the symbol clock.

2 Problem statement

Let $S = S_1...S_K$ be the sequence of quantized source symbols taking their values in a finite alphabet composed of 2^q symbols. The sequence $S_1 \dots S_K$ is assumed to form a Markov chain. Each symbol S_k is then coded into the codeword $\bar{U}_k$ by means of a variable length code. The concatenation of codewords yields the sequence $U = U_1...U_N$ of useful bits, where the length N is a random variable, function of S. K and N are assumed to be known at the receiver. The bitstream U is then fed to the channel coder (punctured syst. conv. code), which yields the sequence $R = R_1...R_M$ of redundant bits. Notice that all randomness comes from S, since U and R are deterministic functions of S. The bitstream (U, R) is sent over a memoryless channel and received as measurements (Y, Z); so the problem consists in estimating S given the observed values $y = y_1...y_N$ and $z = z_1...z_M$, pointwise measurements on useful and redundant bits, respectively.

3 Iterative source-channel decoder

We first build a joint model for the pair Markov source (MS) + source coder (SC). We then design an iterative "turbo" algorithm for joint decoding, which alternates use of this joint model and of the channel coder (CC) model.

3.1 Joint model of the pair MS + SC

Let us first assume that $S_1 \dots S_K$ are i.i.d. with distribution P_s. For a non dyadic P_s, the best variable length code (the Huffman code) has an average codeword length that remains strictly above the lower bound $H(S_i)$ (at most 1 bit above it). Therefore there remains some correlation between the bits at the output of the source coder, that can be exploited to help the segmentation + estimation task at the receiver. This correlation can be modeled by a bit clock HMM obtained by mapping the distribution P_s on the codeword tree (fig. 2). This tree is composed of 2^q leaves (one per symbol), and $2^q - 1$ inner vertices. The state X_n of the model is a vertex ν of the tree, starting at $X_0 = rootnode$, and transitions from X_n to X_{n+1} determine the value of bit U_n. Of course, each leaf of the tree is identified to the rootnode, in order to prepare for the next symbol.

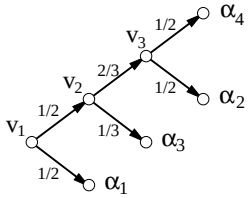

FIG. 2: *Codetree for symbols* $\alpha_1 \dots \alpha_4$ *with respective probabilities* $(\frac{1}{2}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6})$ *and codewords* $(0, 110, 10, 111)$*, assuming* $\nearrow$ *produces bit 1 and* $\searrow$ *produces 0.*

The resulting HMM is represented above the interleaver on fig. 3. A constraint (or perfect observation) can be added on the last state X_N to ensure that the chain finishes at a leafnode, i.e. produces an integer number of symbols. However, to ensure also that the right number K of symbols is obtained, one must extend the state variable X_n with a symbol counter K_n augmented each time a new leaf is reached. With this extended state (X_n, K_n), the constraint at time N can guarantee that only sequences of K symbols are compared. The price to pay is a bigger state space.

Despite the help of these constraints, this inner codeword redundancy is quite low. The symbol correlation due to the MS model represents the major part of the redundancy. To incorporate this information, one must consider that the symbol S_k is actually not produced following the stationary law P_s, but following $P(S_k|S_{k-1})$. This memory effect is again introduced into the model by state augmentation: we store in X_n the last symbol σ produced, hence X_n is now a pair (σ, ν). Transition probabilities follow accordingly: each $P(S_k|S_{k-1} = \sigma)$ is mapped on the codetree, and the value of σ in X_n selects the appropriate mapping. Of course, the value of σ changes each time X_n reaches a new leafnode.

3.2 Joint source-channel turbo-decoding

To finally incorporate the channel coder (CC), one can follow the same lines: build a state representation of the (for example) convolutional code producing bits $R_1 \dots R_M$ from $U_1 \dots U_N$, with state vector X'_n, and aggregate X_n and X'_n to form a global HMM. This time, the state vector becomes intractable, whence the idea to process the two models MS+SC and CC separately. This is done following the principle of serial turbo codes (fig. 3), provided an interleaver is introduced in the coding chain.

In terms of Bayesian networks, connecting the two chains creates N elementary cycles on the global graph. The interleaver makes these cycles become long. Hence algo-

1. By contrast, recall that [6] is assuming an i.i.d. source, which makes the source model useless and removes the difficulty of dealing with two time indexes.

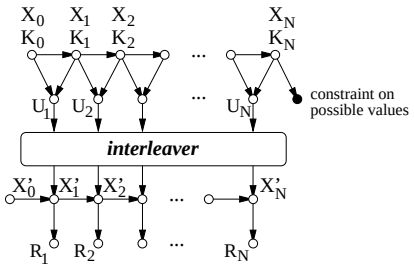

FIG. 3: *Introduction of an interleaver between the HMMs of the pair MS+SC and CC to increase cycle lengths of the Bayesian network.*

rithms theoretically valid on tree-shaped nets (graphs) still perform well since around each variable, the graph is approximately a tree (see [7] and the extended paper [8] for details).

Measurements are available on variables U_n and R_m (not represented on fig. 3). A belief propagation (BP) algorithm can be used to estimate hidden variables $(X, K), U$, X' and R. The organization of message circulations is free, hence it can be performed in two sweeps along X', followed by transmission of (so-called "extrinsic") information towards U and pairs (X, K). Again message circulations can be organized into two sweeps along pairs (X, K) before propagation towards X'. This architecture exactly amounts to computing *a posteriori* probabilities on variables of each model alone, and to iterating between the two models, with exchange of extrinsic information. This exactly reproduces the decoding of serial turbo-codes. At the last step, symbols S_k, or equivalently pairs (X_n, K_n), are estimated with a Viterbi algorithm (sequence MAP), since a simple local MAP on each pair may yield a non valid bitstream. The overall complexity is much smaller than for the aggregated model since few iterations are necessary, on reasonable state dimensions.

4 Splitting the joint model MS+SC

This section goes forward in the direction of separate processing of the models. We now wish to process separately information from the source coder (SC) and from the Markov source (MS). We show that iterations are useless here, and that an interleaver is not needed.

4.1 Symbol clock model for MS+SC

A joint bit-clock model for the pair MS + SC is appropriate for the iterative scheme of the previous section, since the CC provides extrinsic information on bits U_n (or sets of bits). Nevertheless, a symbol clock model is instructive. It takes the form depicted on fig. 4. The Markov chain S_k is displayed, together with the corresponding codewords $\bar{U}_k$ and measurements $\bar{Y}_k$. For the same reasons as above, the state S_k is extended with a bit counter N_k defined as the number of bits in $\bar{U}_1 \dots \bar{U}_k$: $N_k = \mathcal{L}(\bar{U}_1 \dots \bar{U}_k)$ hence the pairs (S_k, N_k) is still a Markov chain. A constraint on the value of the counter N_k at symbol time $k = K$ guarantees that the model assigns a null likelihood to symbol sequences that do not correspond to N bits.

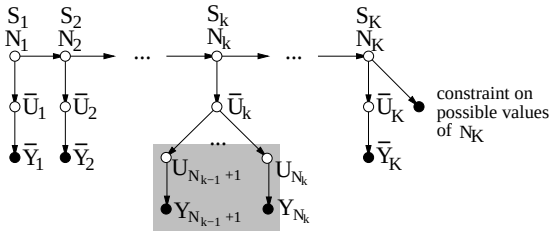

FIG. 4: *Symbol clock model of the pair MS+SC.*

The symbol clock model and the bit clock model are related in a simple manner. Let us represent the expanded trellis of the bit clock model (fig. 5). The state value at time n is composed of a number of symbols k and a pair (σ, ν), so a node of the trellis corresponds to a 4-tuple (n, k, σ, ν). By grouping nodes with the same value k on the second component, one exhibits values of the symbol clock model at time k. This representation is very useful to translate posterior likelihoods on states of the bit clock model to states of the symbol clock model, and conversely.

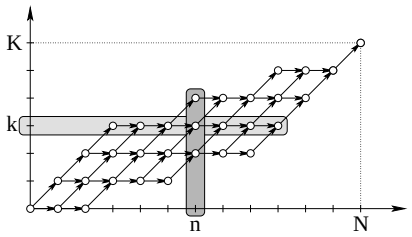

FIG. 5: *Trellis of the joint model for MS+SC. Horizontal axis = bit clock, vertical axis = symbol clock.*

4.2 Separate processing of MS and SC

The graph of the symbol clock model for the pair MS+SC (fig. 4) is a tree, hence estimation of symbols from Y can readily be performed on this graph by a BP algorithm or a Viterbi algorithm. One structure for BP can be designed in the following way : one first estimates codewords $\bar{U}_k$ from measurements $\bar{Y}_k$, which gives estimates on pairs (S_k, N_k), and then forward and backward propagations are performed on the horizontal part to provide the right posteriors $P(S_k, N_k|Y)$. This exactly amounts to 1/ estimating pairs (S_k, N_k) assuming they are independent, and then 2/ taking the intersymbol correlation into account. In other words, this is a way of using separately the two models of the MS and the SC, since independence of symbols is first assumed.

Notice a particular feature of this procedure. The graph of fig. 4 is actually a *random* tree, since the position of measurements $\bar{Y}_k$ depends on the value of variable N_k. However, one can prove that BP algorithms on random trees remain valid, up to some extra technical difficulties [8].

Naturally, one would like to estimate the bitstream $\bar{U}_1 \dots \bar{U}_K$ using a bit clock model, in order to incorporate easily extrinsic information from the CC. This is done in the following way : a bitclock model is designed for SC assuming symbols are independent. This model works with a small state space since X_n doesn't need to store the

last symbol produced (σ). Nevertheless, counter K_n remains. BP is first run on this SC model, then posteriors on (X_n, K_n) are translated into posteriors at the symbol clock, i.e. on pairs (S_k, N_k). Notice that these posteriors are "biased" in some sense because the bitclock model assumes independent S_k's, the N_k's are related one to the next. But this bias can be easily corrected, to provide the true $P(S_k, N_k|\bar{Y}_k)$ which serve as input to BP on the MS model. Final posteriors on $P(S_k, N_k|Y)$ can then be translated back into posteriors on bits U_n to prepare for the next iteration of the turbo procedure.

In summary, up to some technicalities due to the presence of two time indexes, the pair MS+SC can be processed separately. The final algorithm appears as a turbo procedure incorporating in alternance redundancy due to the MS model and to the CC, the intermediary SC model playing the part of a translator of soft information between bits and symbols. More details on algorithms can be found in [8].

5 Experimental results

The joint decoding procedure has been evaluated on a first-order Gauss-Markov source, with zero-mean, unit-variance and correlation factor $\rho = 0.9$. The source is quantized with a 4-bit uniform quantizer on the interval $[-3, 3]$, and we consider sequences of $K = 200$ symbols. The VLC source coder is based on a Huffman code, designed for the stationary distribution of the source. The channel code is a rate $1/2$ recursive systematic convolutional code with transfer function $F(z)/G(z)$, $F(z) = 1 + z + z^2 + z^4$, $G(z) = 1 + z^3 + z^4$. Since very few errors have been observed with rate $1/2$, the code was punctured to rate $3/4$. A variable size interleaver is introduced between the source coder and the channel coder. All simulations have been performed assuming an AWGN channel.

Figure 6 provides the residual bit error rates (BER) and symbol error rates (SER) for different channel E_b/N_0. On each plot, the top curve corresponds to an ML estimation of the bitstream assuming independent bits (and no channel coding), followed by a hard Huffman decoding. On the BER plot, the second curve corresponds to a MAP channel decoding, assuming an input of independent bits. The third one is the result of the first iteration, where knowledge on symbol correlation and codeword structure has been introduced. Successive curves show the extra gain of iterations in the procedure, which depends on the degree of redundancy present on both sides of the source coder. In particular, when a white source is assumed, little gain is obtained through iterations, since little information is present on the left hand side of the interleaver : mainly the remaining intra-symbol redundancy, and constraints on K and N.

6 Conclusion

The turbo principle, revealed by turbo codes, can be generalized into the iterative use of factors of a big product model. It is a promising strategy that has improved existing estimation algorithms in many problems, at almost no cost. We have shown that this strategy performs well in this specific problem of joint source-channel decoding. However, its use is not always relevant : in the particular case of the product model of the source + source coder, one doesn't need to separate factors by an interleaver. The iterative use of the factors can be optimal. This advocates a careful understanding of dependencies before choosing a turbo strategy. Using the three models separately allows to tune one component without dramaticaly changing the decoding algorithm. For example, the variable length source coder can be made more robust to fight against desynchronizations at low SNRs (see the companion paper [9] in this conference).

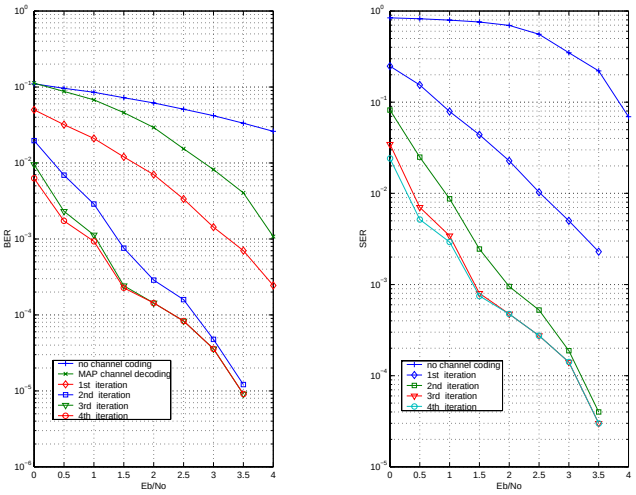

FIG. 6: *Residual BER (left) and SER (right) vs E_b/N_0, for successive iterations, with a Gauss-Markov source.*

Références

[1] D. J. Miller and M. Park, "A sequence-based approximate MMSE decoder for source coding over noisy channels using discrete hidden markov models," *IEEE Trans. on Communications*, vol. 46, no. 2, pp. 222–231, February 1998.

[2] M. Park and D.J. Miller, "Decoding entropy-coded symbols over noisy channels by MAP sequence estimation for asynchronous HMMs," in *Proc. Conf. on Info. Sciences and Systems*, May 1998.

[3] A.H. Murad and T.E. Fuja, "Joint source-channel decoding of variable length encoded sources," in *Proc. Information Theory Workshop, ITW*, June 1998, pp. 94–95.

[4] N. Demir and K. Sayood, "Joint source-channel coding for variable length codes," in *Proc. IEEE Data Compression Conference, DCC*, March 1998, pp. 139–148.

[5] R. Bauer and J. Hagenauer, "Turbo-fec/vlc-decoding and its applications to text compression," in *Proc. Conf. on Information Sciences and Systems*, March 2000.

[6] R. Bauer and J. Hagenauer, "Iterative source/channel decoding based on a trellis representation for variable length codes," in *Proc. Int. Symp. on Information Theory, ISIT*, June 2000, p. 238.

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

[8] A. Guyader, E. Fabre, C. Guillemot, and M. Robert, "Joint source-channel turbo decoding of entropy-coded sources," *to appear in IEEE Sel. Areas in Com.*, 2001.

[9] A. Guyader, E. Fabre, and C. Guillemot, "Robust decoding of VLC encoded Markov sources," in *Proc. GRETSI*, sept. 2001.