Multiple parallel concatenation of circular recursive systematic convolutional (Crsc) codes

Claude Berrou, Catherine Douillard, Michel Jezequel

Multiple parallel concatenation of circular recursive systematic convolutional (CRSC) codes

Claude BERROU, Catherine DOUILLARD and Michel JEZEQUEL*

*ENST Bretagne, Electronics Department, BP 832, 29285 Brest Cedex, France

Abstract

This paper deals with a family of error correcting codes based on circular recursive systematic convolutional (CRSC) codes. A multidimensional parallel concatenation of CRSC codes is proposed to reach minimum distances close to those of random codes. For information blocks of size k, minimum distances as large as $k/4$ may be obtained if the code dimension is brought up to 4 or 5. Such codes can be decoded using an iterative (turbo) process relying on extrinsic information concept.

Keywords : convolutional codes, random coding, code concatenation, extrinsic information, turbo code

Résumé

Cet article présente une famille de codes correcteurs d'erreurs bâtie autour de codes convolutifs récursifs systématiques circulaires (CRSC). Une concaténation parallèle multi-dimensionnelle de codes CRSC est utilisée pour atteindre des distances minimales proches de celles qui sont obtenues avec les codes aléatoires. Pour une taille k de bloc d'information, des distances minimales proches de $k/4$ peuvent être ainsi obtenues lorsqu'on porte la dimension du code à une valeur de 4 ou 5. De tels codes peuvent être décodés en utilisant une procédure itérative ("turbo") s'appuyant sur le concept d'information extrinsèque.

Mots clés : codes convolutifs, codes aléatoires, concaténation, information extrinsèque, turbo code

Contents

I. INTRODUCTION

Since the Shannon pioneering work [1], random coding has always represented a reference for error correction. Systematic random encoding of k information bits, providing n-bit codewords, requires firstly to draw at random k (n - k)-bit « markers », and to store them in a memory, where the storage address is i ($1 \le i \le k$). Secondly, the redundancy of any k-bit information block is calculated by modulo-2 adding all the markers whose addresses i correspond to the places of logic '1' in the information block. The final codeword consists of the concatenation of the k information bits and the $n-k$ redundancy bits. The code rate R is equal to k/n. This very simple way of

building up codewords gives a linear code and leads to large minimum distances for values of $n-k$ great enough. Since two codewords may only differ on one information bit, and since the redundancy is chosen at random, the statistical minimum distance is equal to $1+\frac{n-k}{2}$. However, as the minimum distance of this code is a random variable, its different realizations may be smaller than this value. Appendix A provides the expression of the probability that minimum distance $d_{\min}$ is greater than or equal to a given value *D*.

For usual values of *n* and *k*, decoding random codes is not practically feasible. The only way of decoding consists in passing the 2^k different codewords in review, and to keep the closest one to the received word (i.e. the most likely codeword). Therefore, decoder complexity increases exponentially with the length *k* of the sequence to encode, and becomes unusable for practical applications.

This paper proposes a code family imitating random codes and presenting moderate decoding complexity. We specially focus on minimum distances achieved by these codes based on a multiple parallel concatenation of recursive systematic convolutional codes. Besides, elementary encoding is organized so as to perform block encoding, due to the circular coding concept, which is developed in the next chapter.

II. CIRCULAR CONVOLUTIONAL CODES

Convolutional codes are not *a priori* really suited to encode information transmitted in block form. The knowledge of the initial state of the trellis is not a problem, as the« all zero » state is, in general, forced by the encoder. However, the decoder has no special available information regarding the final state of the trellis. This problem is even more serious for *N*-dimensional turbo codes, since the decoder does not know the final states of the *N* trellises after *N* encoding processes. Several answers can solve this problem, for example:

- **doing nothing**, that is, no information concerning the final states of the trellises is provided to the decoder. The decoding process is less effective for the last encoded data and the asymptotic gain may be reduced. This degradation is a function of the block length and may be low enough to be accepted for a given application.
- **forcing the encoder state** at the end of the encoding phase for one or all dimensions. This solution has been adopted by the CCSDS standard [2]. Tail bits are used to « close » the trellises and then sent to the decoder. This method presents two major drawbacks. First, minimum weight w_{min} is no more equal to 2 for all information data[1], since, at the end of each block, the second '1' bringing the encoder back to the « all zero » state may be a part of the tail bits. In that case, turbo decoding is handicapped if tail bits are not encoded at least twice. Next, the spectral efficiency of the transmission is degraded and the cost is all the higher since blocks are short and *N* is high.

[1] The weight *w* of a binary word is defined as the number of information bits equal to '1', that is the number of information bits differing from the « all zero » word, which is used as a reference for linear codes. For a recursive code, when the final states are fixed by the encoder, the minimum value for *w* is 2. For more details see [3-5], for example.

– adopting **circular coding**. For circular convolutional codes, the encoder is left in the initial state at the end of the encoding stage. Decoding trellis can therefore be seen as a circle and decoding may be initialized everywhere on this circle. This technique, well known for non recursive codes, has been adapted to the specificity of recursive codes.

II.1. Principle of circular recursive systematic convolutional (CRSC) codes

Let us consider a recursive convolutional encoder, for instance Figure 1 encoder (quaternary code with memory ν equal to 3). At time *i*, register state $\mathbf{S}_i$ is a function of previous state $\mathbf{S}_{i-1}$ and input vector $\mathbf{X}_i$. Let $\mathbf{G}$ be the generator matrix of the considered code. States $\mathbf{S}_i$ and $\mathbf{S}_{i-1}$ are linked by the following recursion relation:

(1) $\mathbf{S}_i = \mathbf{G}.\mathbf{S}_{i-1} + \mathbf{X}_i$

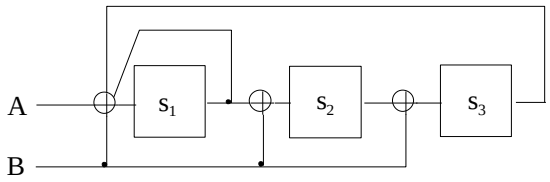

Figure 1. Recursive convolutional (quaternary) encoder with memory $\nu = 3$

For Figure 1 encoder, vectors $\mathbf{S}_i$ and $\mathbf{X}_i$, and matrix $\mathbf{G}$ are given by:

$$\mathbf{S}_i = \begin{bmatrix} s_{1,i} \\ s_{2,i} \\ s_{3,i} \end{bmatrix};\ \mathbf{X}_i = \begin{bmatrix} A_i + B_i \\ B_i \\ B_i \end{bmatrix};\ \mathbf{G} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

From (1) we can infer:

$$\mathbf{S}_i = \mathbf{G}.\mathbf{S}_{i-1} + \mathbf{X}_i$$
$$\mathbf{S}_{i-1} = \mathbf{G}.\mathbf{S}_{i-2} + \mathbf{X}_{i-1}$$
$$\mathbf{S}_1 = \mathbf{G}.\mathbf{S}_0 + \mathbf{X}_1$$

Hence, $\mathbf{S}_i$ may be expressed as a function of initial state $\mathbf{S}_0$ and of data applied to encoder input between times 1 and *i*:

(2) $$\mathbf{S}_i = \mathbf{G}^i.\mathbf{S}_0 + \sum_{p=1}^{i} \mathbf{G}^{i-p}\mathbf{X}_p$$

If *k* is the input sequence length, it is possible to find a state $\mathbf{S}_c$ so that $\mathbf{S}_c = \mathbf{S}_k = \mathbf{S}_0$. Its value is derived from (2):

(3) $$\mathbf{S}_c = \left\langle \mathbf{I} + \mathbf{G}^k \right\rangle^{-1}.\ \sum_{p=1}^{k} \mathbf{G}^{k-p}\mathbf{X}_p$$

where $\mathbf{I}$ is the identity matrix.

State $\mathbf{S}_c$ exists only if $\mathbf{I}+\mathbf{G}^k$ is invertible[2]. Particularly, k can not be a multiple of the period L of the encoder recursive generator. L is defined as:

$$\mathbf{G}^L = \mathbf{I}$$

$\mathbf{S}_c$ is the conservative encoder state. That is, if the encoder starts from state $\mathbf{S}_c$, it comes back to the same state when the encoding of the k data (k couples for Figure 1 encoder) is completed. The value of $\mathbf{S}_c$ is a function of the data sequence. Such an encoding process is called circular because the associated trellis may be viewed as a circle, without any discontinuity on transitions between states.

Determining $\mathbf{S}_c$ requires an pre-encoding stage. First, the encoder is initialized in the « all zero » state $\mathbf{S}_0$. Then, the data sequence of length k is encoded once, leading to final state $\mathbf{S}_k^0$. Thus, from (2):

$$\mathbf{S}_k^0 = \sum_{p=1}^{k} \mathbf{G}^{k-p}\mathbf{X}_p$$

Combining this result with (3), the value of conservative state $\mathbf{S}_c$ can be linked to $\mathbf{S}_k^0$ as follows:

$$\mathbf{S}_c = \left\langle \mathbf{I}+\mathbf{G}^k \right\rangle^{-1}.\ \mathbf{S}_k^0 \qquad (4)$$

In a second stage, data are definitely encoded from state $\mathbf{S}_c$.

In practice, the relation between $\mathbf{S}_c$ and $\mathbf{S}_k^0$ is provided by a small combinational operator with ν input bits and ν output bits, if ν represents the code memory. The major disadvantage of this method lies in encoding the sequence twice: once from « all zero » state and once from state $\mathbf{S}_c$. Nevertheless, the encoding stage can be performed at a frequency much higher than the data rate, so as to reduce the encoding latency effects.

II.2. Circular codes and turbo codes

Circular codes are well suited to turbo code encoding and decoding concepts. For example, let us consider the binary turbo encoder in Figure 2. The data sequence to be encoded, made up of k information bits, is applied twice at the CRSC encoder input: first, in the data natural order (switch in position 1), and next in an interleaved order, given by time permutation function Π (switch in position 2). In fact, the circular code principle may be applied in two slightly different ways, according to whether the code is self-concatenated or not.

[2] Note that some matrices **G** are not convenient.

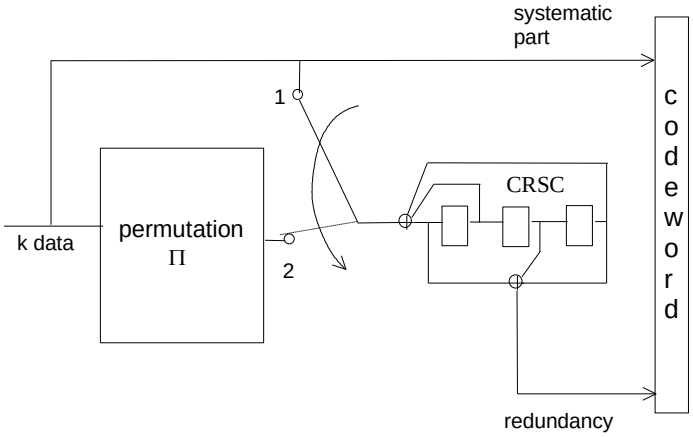

Figure 2. Turbo code built from a CSRC code.

Case 1 : the code is self-concatenated.

The second encoding stage directly follows on the first stage without intermediate reinitializing of the encoder state. Conservative state $\mathbf{S}_c$ is calculated for the whole sequence of length $2k$. At reception, the decoder performs a global decoding of the double length sequence.

Case 2 : the code is not self-concatenated.

The encoder is initialized at the beginning of each encoding stage. Two conservative states $\mathbf{S}_{c1}$ and $\mathbf{S}_{c2}$, corresponding to the both encoded sequences, are calculated. At reception, both sequences of length k are decoded separately.

Depending on the case, data encoding may be represented with one or two circular or « tailbiting » trellises. Whatever elementary algorithm is applied, iterative decoding requires repeated turns around the circular trellis(es), the extrinsic information table being continuously updated during data processing. Iterations naturally follow one from each other without any discontinuity between transitions from state to state.

In the case where the APP algorithm (also called MAP, backward-forward, or BCJR [6] algorithm), or one of its simplified version [7] is applied, decoding the sequence consists in going round the circular trellis anticlockwise for backward process, and clockwise for forward process (Figure 3), where data are decoded and extrinsic information is built. For both processes, probabilities computed at the end of a turn are used as initial values for next recursion. The number of turns performed around the « tailbiting » trellis(es) is equal to the number of iterations required by the iterative decoding process. In practice, the iterative process is preceded by a « prologue » decoding step, performed on a part of the circle for a few ν. It is intended to « guide » the process towards an initial state which is a good estimation of the conservative state.

If the decoding algorithm is a soft-output version of the Viterbi algorithm, iterative decoding requires the circular trellis to be processed anticlockwise for metrics computation, with a partial clockwise return to calculate decisions and extrinsic information. As for APP algorithm, iterations naturally follow one from each other, and the metrics computed at the end of each turn are used as initial metrics for the next iteration. An estimation of the conservative state may also be obtained by a prologue step.

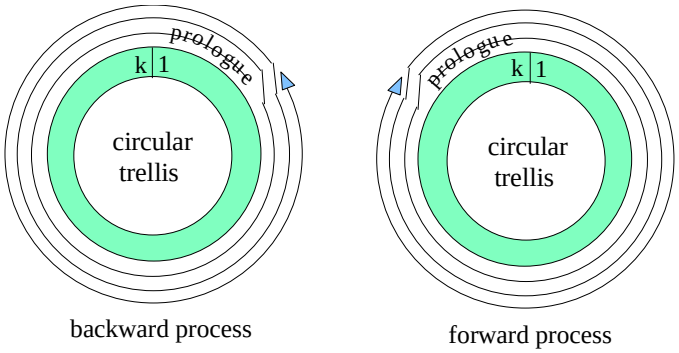

Figure 3. Circular code processing with backward-forward algorithm .

III. MULTIDIMENSIONAL CONCATENATION OF CRSC CODES

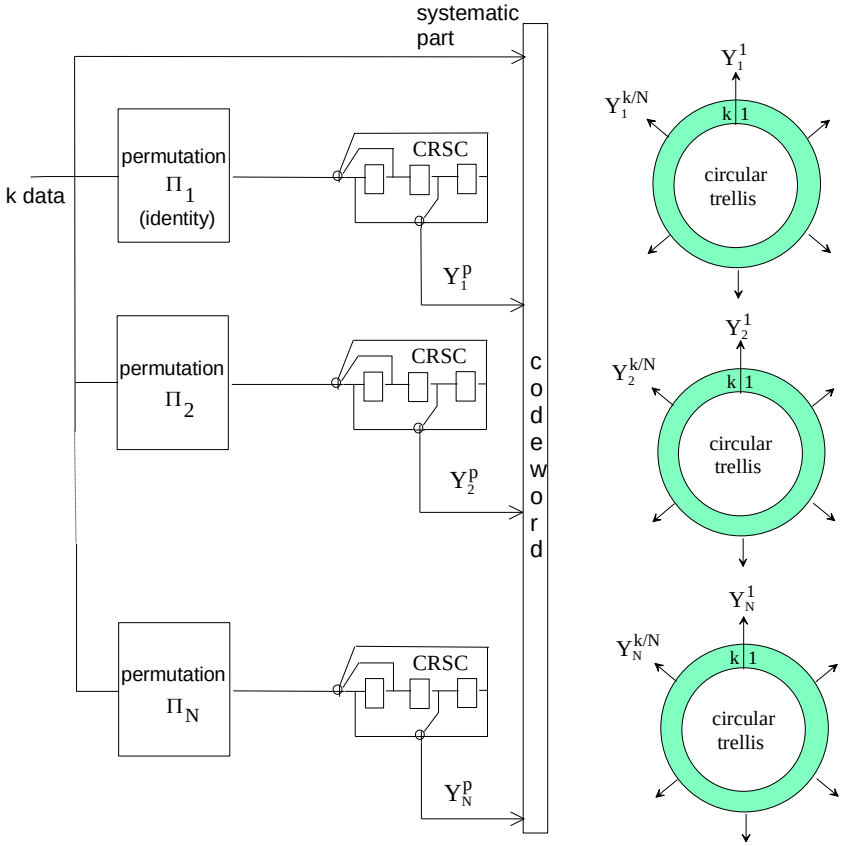

Figure 4. Parallel concatenation of N CRSC codes

Let us consider the encoding process of a k-data block, using the (a ?) parallel concatenation of N CRSC codes in a similar way to a standard turbo code [3]. The global code rate is supposed to be 1/2. Figure 4 represents the global encoder and the N corresponding circular trellises. Permutations Π_j $(1 \le j \le N)$ are chosen randomly, except the first one which is the identity permutation. Each elementary encoder provides k / N redundancy bits Y_j^p $(1 \le j \le N, 1 \le p \le k / N)$, regularly emitted during the sequence encoding process. The study of distance properties of this multiconcatenated code is not an obvious problem in general and would

not give any practical interesting information. However, three particular cases, corresponding to values of N equal to 1, 2, and k, can be considered. For $N = 1$, we have to deal with a simple convolutional code ; for $N = 2$, the code is a classic turbo code. These two particular cases have already been widely examined, especially in [11] for a turbo code using a random permutation.

The extreme case $N = k$, where each elementary encoder provides only one redundancy symbol (bit ?) Y_j^1, behaves very close to random coding. Let us assume that the redundancy symbol is calculated when the sequence encoding starts, that is when the encoder is in the conservative state $\mathbf{S}_c$. Then, two cases have to be examined, according to the value of the weight w of the sequence to be encoded.

$w = 1$. Relation (3) shows that the conservative state $\mathbf{S}_c$ can never be the « all zero » state (besides, the encoder never goes to the « all zero » state during the encoding process of the sequence, whatever the permutation). The redundancy bit may be '0' or '1' with equal probability.

$w \geq 2$. The conservative state may be any state. The probability that $\mathbf{S}_c$ is different from the « all zero » state is $\frac{2^\nu - 1}{2^\nu}$. Consequently, the probability that the redundancy symbol is equal to '1' is $\frac{1}{2}.\frac{2^\nu - 1}{2^\nu}$. For values of ν great enough (let us say $\nu \geq 3$), it can be supposed that both logical values are almost equiprobable.

Thus, the code redundancy is made up of a set of values Y_j^1 $(1 \leq j \leq k)$ comparable to a set of binary random values: the value of each redundancy symbol Y_j^1 depends on $\mathbf{S}_c$, which is directly linked to the permutation implemented at place j of the concatenated encoding process, and this permutation is drawn (hit ?) at random. Hence this code must show minimum distances of the same order as those allowed by probabilities given by relation (A-4). In order to confirm this assertion, we programmed, for different dimensions, three multiconcatenated codes associated to three values of k retained as examples in Figure A-1: 40, 60, and 80. The elementary encoders use the same generator polynomials 15, 17 ($\nu = 3$). The program tried to obtain minimum distances as great as possible[3], for each case and by trying different permutations. Results are presented in Figure 5. We can notice that minimum distances $d_{\min}$ increase with dimension N, but do not exceed the values predicted by the curves in Figure A-1, that is round about $k/4$. Even when performing an exhaustive search with a more powerful computer, the probabilities that we get maximum values greater than those actually obtained would be very small. Besides, the maximum values have been obtained for dimensions much lower than maximum dimension k, i.e. in the order of 4 or 5 for the cases considered. In consequence, we do not need to use maximum dimension codes to get large values for $d_{\min}$. Much lower dimensions than k provide multiconcatenated codes with features comparable to those of random codes.

[3] The search, which can last several days on an Ultra Sparc computer, was performed for weight values up to $w = 8$. For the cases considered, taking greater weights into account does not seem to influence our results, but we can not prove it.

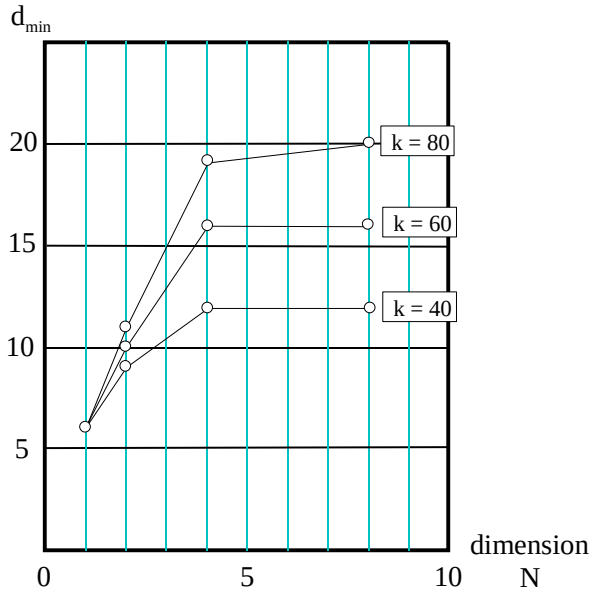

Figure 5. Minimum distances as a function of the dimension of the code, N, for k = 40, 60, 80. Results obtained by computer searching.

Multiconcatenated codes may be decoded using the « turbo » principle. Each elementary decoder in place j $(1 \le j \le N)$ provides for each binary information data d_i $(1 \le i \le k)$, an extrinsic information symbol Z_j^i, as a probability or a LLR (Logarithm of Likelihood Ratio). Each decoder in place j' takes advantage of the work performed by the $N-1$ other decoders which transmit it their extrinsic information $Z_{j \ne j'}^i$. At each decoder input, the different extrinsic information symbols relating to the same data are simply combined through a product, for probabilities, or a sum, for LLR.

However, turbo decoding is not optimal [12], so it is quite possible that, for great values of N (i.e. for elementary encoders with low redundancy rate) the loss due to iterative decoding is increased. Thus, it would not be possible to take a complete advantage of the error-correcting capability of large dimension codes. Fortunately, we showed that, with dimension values of 3, 4, or 5, we obtain distances close to maximum values and that turbo decoding suboptimality does not seem to be a real handicap (e.g., see [13] for a three-dimensional turbo code).

IV. CONCLUSION

We showed that a maximum dimension concatenation of CSRC codes may be compared to random coding. Hence, minimum distances of such codes are large (great ?), in the order of the quarter of the block length. In fact, an experiment with some values of k led us to observe that these values of minimum distance could be obtained for dimensions much lower than k, but slightly greater than 2, the dimension of a standard turbo code.

Two-dimensional turbo codes can be penalized because of insufficient minimum distances for certain applications, especially for short blocks when very low bit error rate is required. A possible answer to this problem consists in increasing the code dimension, practically up to 3, 4, or 5. In

return, hardware complexity is raised by 50, 100 or 150 %, for the same decoding latency. For bit error rates greater than 10^{-5} typically, increasing the dimension of the turbo code beyond $N = 2$ does not give extra gain.

From a more conceptual point of view, a remark can be made on the codes presented in this paper in relation to Shannon statistical approach. When establishing his famous theorems, he considered a set of random codes and he predicted average statistical performance results, arguing that, amongst all the possible codes, there is at least one which shows performance equal or higher than average. Now, considering a multiple parallel concatenation of CRSC codes, and especially in the extreme case $N = k$, we can take the advantage of whole error-correcting capability only if **all the codes are actually used, and not only the best of them** (besides, we would be unable to give a definition for it). Nowadays, the « turbo » technique allows a great number of codes to be decoded, and in fact, it appears to be a indirect means of taking advantage of the properties of a code statistically good, without being having to search for the best one amongst all the possibilities.

APPENDIX A

The purpose of this appendix is to calculate the probability $\Pr\{d_{\min} \geq D\}$ that a codeword produced by random encoding of *k*-bit data blocks, with code rate 1/2, has a minimum distance $d_{\min}$ greater than or equal to *D*. The principle of random coding is described in section I.

As the code is linear, distances may be determined relatively to the « all zero » word. The distance *d* of a codeword may be expressed as the sum of the weights (that is, the number of logical '1') of the systematic part *w* and of the redundancy d_r.

(A-1) $d = w + d_r$

For a given weight *w*, $\binom{k}{w}$ redundancy sequences or sequence combinations (obtained by modulo-2 additions) have to be considered. The probability that one of these sequences or sequence combinations contains at least $D - w$ '1' is:

(A-2) $$\Pr\{d_r \geq D - w \,/\, \text{one combination of weight } w\} = 2^{-k} \sum_{i=D-w}^{k} \binom{k}{i}$$

The probability that all the redundancy sequences or combinations contain at least $D - w$ '1' is:

(A-3) $$\Pr\{d_r \geq D - w \,/\, \text{all combinations of weight } w\} = \left[2^{-k} \sum_{i=D-w}^{k} \binom{k}{i} \right]^{\binom{k}{w}}$$

In order that all the codewords have a distance greater than or equal to *D*, the above relation has to be multiplied by itself for all the possible values of *w*, from 1 to $D-1$. Therefore,

$$\Pr\{d_{\min} \geq D\} = \prod_{w=1}^{D-1} \left[2^{-k} \sum_{i=D-w}^{k} \binom{k}{i} \right]^{\binom{k}{w}}$$

This relation may also be expressed as:

$$\text{(A-4)} \quad \Pr\{d_{\min} \geq D\} = \prod_{w=1}^{D-1} \left[1 - 2^{-k} \sum_{i=0}^{D-w-1} \binom{k}{i} \right]^{\binom{k}{w}}$$

This probability is plotted in Figure A1 as a function of D for six values of k, from 40 to 140. It can be noticed that a minimum distance $d_{\min}$ in the order of $k/4$ can be easily reached, but because of the steep (abrupt ?) decrease of the probabilities, it seems unrealistic to get distances beyond this typical value, even with very powerful computers which would try to adapt redundancy patterns so as to raise $d_{\min}$. For a more theoretical approach, see for instance [14-16].

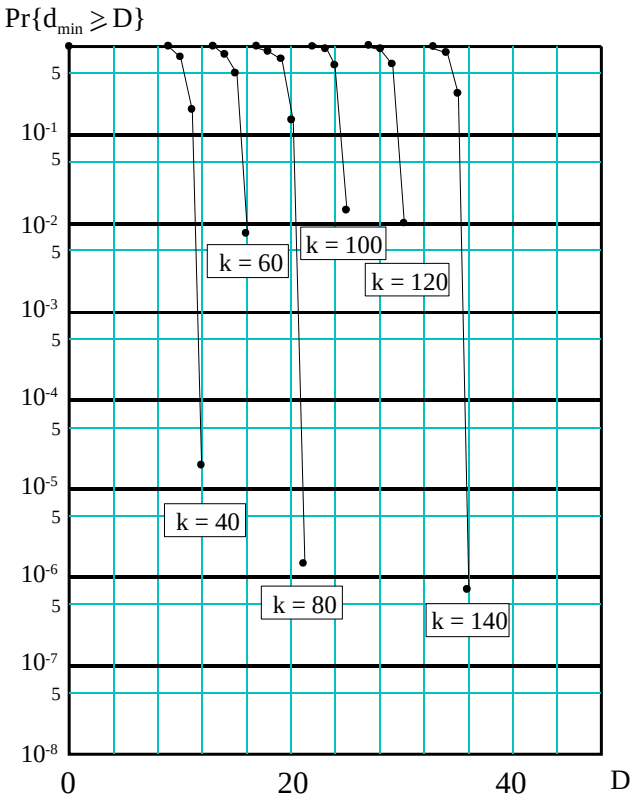

Figure A1. Probability that random coding of a k-bit data word leads to minimum distance $d_{\min}$ greater than or equal to D.

Acknowledgements

The authors wish to thank Professors Alain Glavieux (ENST Bretagne) and Shlomo Shamai (Shitz) for their useful help in writing this paper.

REFERENCES

[1] C. E. Shannon, "A Mathematical Theory of Communication," Bell System Technical Journal, Vol. 27, July and October 1948.
[2] "Draft CCSDS recommendation for telemetry channel coding (updated to include turbo codes)", Consultative Committee for Space Data Systems, Rev. 4, May 1998.
[3] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes", *IEEE Trans. Com*, Vol. 44, N° 10, pp. 1261-1271, Oct. 1996.
[4] C. Berrou and A. Glavieux, "Turbo-codes, general principles and applications", Proc. of the 6th Int. Tirrenia Workshop on Digital Communications, Pisa, Italy, pp. 215-226, Sept. 1993.
[5] P. Thitimajshima, "Les codes convolutifs récursifs systématiques et leur application à la concaténation parallèle" (in French), Ph. D. N° 284, Université de Bretagne Occidentale, Brest, France, Dec. 1993.
[6] L.R. Bahl, J. Cocke, F. Jelinek and J. Raviv : "Optimal decoding of linear codes for minimizing symbol error rate", IEEE Trans. Inform. Theory, **IT-20**, pp. 248-287, Mar. 1974.
[7] P. Robertson, P. Hoeher and E. Villebrun, "Optimal and suboptimal maximum a posteriori algorithms suitable for turbo decoding", European Trans. Telecommun., vol. 8, pp. 119-125, March-Apr. 1997.
[8] G. Battail, "Pondération des symboles décodés par l'algorithme de Viterbi" (in French), Ann. Télécommun., Fr., 42, N° 1-2, pp. 31-38, Jan. 1987.
[9] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications", Proc. of Globecom '89, Dallas, Texas, pp. 47.11-47-17, Nov. 1989.
[10] C. Berrou, P. Adde, E. Angui and S. Faudeil, "A low complexity soft-output Viterbi decoder architecture", Proc. of ICC '93, Geneva, pp. 737-740, May 1993.
[11] S. Benedetto and G. Montorsi, "Unveiling turbo-codes: some results on parallel concatenared coding schemes", *IEEE Trans. I. T.*, vol. 42, N° 2, pp. 409-429, Mar. 1996.
[12] C. Berrou, "Some clinical aspects of turbo codes", Proc. of Int. Symposium on Turbo Codes, Brest, France, pp. 26-31, Sept. 1997.
[13] Li Ping, "Modified turbo codes with low decoding complexity", Elect. Letters, vol. 34, N° 23, pp. 2228-2229, Nov. 1998.
[14] J. N. Pierce, "Limit distribution of the minimum distance of random linear codes", *IEEE Trans. I. T.*, vol. IT-13, N° 4, Oct. 1967.
[15] Y. V. Svirid, "Weight distributions and bounds for turbo codes", Eur. Trans. On Telecomm., vol. 6, N° 5, pp. 543-555, Sept./Oct. 1995.
[16] G. Battail, *Théorie de l'information, application aux techniques de communication* (in French), Masson, Paris, 1997.