Advanced Error-Control Coding Methods Enhance Reliability of Transmission and Storage Data Systems

Karel VLČEK

Dept. of Informatics, Technical University of Ostrava, 17. listopadu 15, 708 33 Ostrava-Poruba, Czech Republic

karel.vlcek@vsb.cz

Abstract. *Iterative coding systems are currently being proposed and accepted for many future systems as next generation wireless transmission and storage systems. The text gives an overview of the state of the art in iterative decoded FEC (Forward Error-Correction) error-control systems. Such systems can typically achieve capacity to within a fraction of a dB at unprecedented low complexities. Using a single code requires very long code words, and consequently very complex coding system. One way around the problem of achieving very low error probabilities is turbo coding (TC) application. A general model of concatenated coding system is shown - an algorithm of turbo codes is given in this paper.*

Keywords

Parallel concatenated coding, iterative decoding, code performance, Error-Control Coding system reliability.

1. Introduction

The principle of modern error-control codes based on the paradigm of turbo codes is to use two codes that are not very long in a manner that reduces the complexity considerably without increasing the error rate significantly. The iterative decoding deals with the distributions of extrinsic information. Produced by the iteration of decoding, the trade-offs between performance and complexity become exact analytical results for simple decoders as well as linear time coding techniques.

A simple derivation for the performance of turbo codes, and a straightforward presentation of the iterative decoding algorithm are included. The derivations of the performance are novel. The treatment is intended for further study in the field and, significantly, to provide sufficient information for the VHDL simulations of design.

The invention of turbo codes involved reviving some concepts [19] and algorithms, and combining them with some clever new ideas [6], [15], [7], [18], [3]. Because the principles surrounding turbo codes are both uncommon and novel, it has been difficult for the initiate to enter into the study [21] of convolutional codes.

Another complicating matter is the fact that now there are numerous papers on the topic so that there is no clear place to begin studying of these codes. This paper is addressed to this problem by including an introduction to the study of turbo codes in one paper [1], [4].

The paper borrows from some publications, sometimes adding details that were omitted in those works. However, the general presentation and some of the derivation are novel. Our goal is self-contained, simple introduction to turbo codes for those already knowledgeable in the fields of algebraic and trellis codes.

The paper focuses especially on the turbo codes. As channel models the binary symmetric channel, the additive white Gaussian noise as well as channels with memory are considered. A detailed description of the encoder is given [16] and a simple derivation of the performance in additive white Gaussian noise (AWGN) is presented.

The understanding and simulation of the iterative decoding algorithm is particularly difficult for the novice, and so we give a thorough description of the algorithm here [18], [9]. The analysis of iterative decoding systems gives the sense how one can use this knowledge for improving them even further.

The subsequent section then describes the iterative algorithm used for decoding of turbo codes. The treatment in each of these sections is meant to be sufficiently detailed so that one may design a computer simulation of the encoder and decoder with reasonable ease. In more details, the paper describes analytic properties of circuit models and practical issues.

A turbo encoder is a combination of two simple encoders. The input is a block of K information bits. The two encoders generate parity symbols from two simple recursive convolutional codes, each with a small number of states. The information bits are also sent in the non-coded form.

2. Encoder and Decoder

The key innovation of turbo codes is an interleaver function P, which permutes the original information bits before input to the second encoder. The permutation *P* allows that input sequence for which one encoder produces low-weight code words will usually cause the other encoder to produce high-weight code words. Thus, even though the constituent codes are individually weak, the combination is surprisingly information bits powerful.

The resulting code has features similar to "*random*" block code with *K=n/k* ratio. Random block codes are known to achieve Shannon-limit performance as *K* gets large hardware as well as the price of a prohibitively complex decoding algorithm. Turbo codes minimize it to good random codes (for large *K*) using an iterative decoding algorithm based on simple individually matched to the simple constituent codes.

Each constituent decoder sends information of the decoded bits to the other decoder, and uses the corresponding from the other decoder as *à priori* likelihood. The non-coded information bits are available to each decoder to initialize the *a priori* likelihood.

The decoder "*MAP*" (*Maximum a Posteriori*) bitwise decoding algorithm requires the states as well known Viterbi algorithm. The turbo decoder iterates between two constituent decoders to reach satisfactory convergence. The final output is one of the likelihood estimates the either of the decoders [10].

In addition to providing improved performance, turbo decoders [17] are lower in complexity of concatenated decoders. Decoding time is proportional to the number of iterations, unless special-purpose hardware is used for parallel-processing all of the states. Interleaver size impacts buffer requirements of decoding time: it is a primary determinant of turbo code performance is achieved with interleavers of the traditional concatenated systems.

As it can be seen in Fig. 1, a turbo encoder consists of two binary rate 1:2 convolutional encoders separated by an *N*-bit interleaver or permuter, together with an optional puncturing mechanism. Clearly, without the puncture, the encoder is rate 1:3, mapping *N* data bits to 3*N* code bits.

The encoders are configured in a manner reminiscent of classical concatenated codes. However, instead of cascading the encoders in the usual serial fashion, the encoders are arranged in a so-called *parallel concatenation*. Observe also that the constituent convolutional encoders are of the recursive systematic variety.

Because any non-recursive (i.e., feed-forward) non-catastrophic convolutional encoder is equivalent to a recursive systematic encoder in that they possess that same set of code sequences, there was no compelling reason in the past for favoring recursive encoders. The recursive encoders are necessary to attain the exceptional performance provided by turbo codes. Without any essential loss of generality, we assume that the constituent codes are identical.

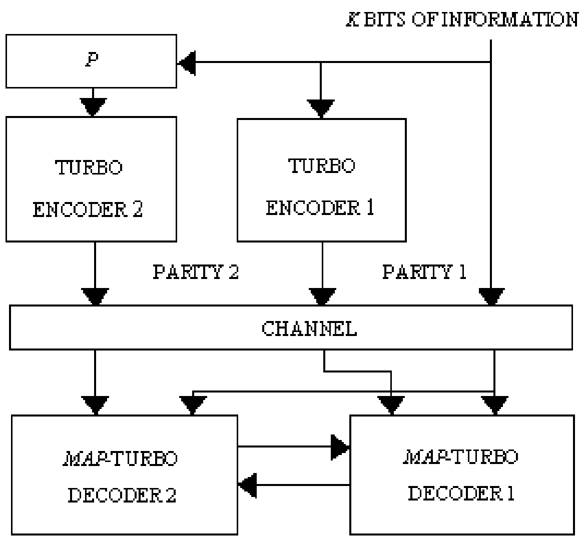

Fig. 1. Standard turbo-coding system

3. Systematic Turbo Encoders

Whereas the generator matrix for a rate 1:2 Non-Recursive convolutional code has the form

$$G_{N-R} = \left[g_1(D) \quad g_2(D) \right],$$

the equivalent Recursive systematic encoder has the generator matrix

$$G_R = \left[1 \quad \frac{g_2(D)}{g_1(D)} \right].$$

Observe that the code sequence corresponding to the encoder input *u*(*D*) for the former code is

$$u(D) \cdot G_{N-R} = \left[u(D) \cdot g_1(D) \quad u(D) \cdot g_2(D) \right],$$

and that the identical code sequence is produced in the recursive code by the sequence

$$u'(D) = u(D) \cdot g_1(D),$$

since in this case the code sequence is

$$u(D) \cdot g_1(D) \cdot G_R = u(D) \cdot G_{N-R}.$$

Here we loosely call the pair of polynomials $u(D).G_{N-R}$ a code sequence, although the actual code sequence is derived from this polynomial pair in the usual way. Observe that, for the recursive encoder, the code sequence will be of finite weight if and only if the input sequence is divisible by $g_1(D)$.

The weight-one input will produce an infinite weight output for such input is never divisible by $g_1(D)$. For any non-trivial $g_1(D)$, there exists a family of weight-two inputs of the form $D^j(1+D^{q-1})$, $j \geq 0$, , which produce finite weight outputs, i.e., which are divisible by $g_1(D)$. When $g_1(D)$. is a primitive polynomial of degree *m*, then $q=2^m$; more generally, q-1 is the length of the pseudo-random sequence generated by $g_1(D)$.

The code word weight-one input will create a path that diverges from the all-zeros path, but never remerge, and there will always exist a trellis path that diverge and remerge later which corresponds to a weight-two data sequence. Consider the recursive code with generator matrix

$$G_R(D)=\left[1 \quad \frac{1+D^2+D^3+D^4}{1+D+D^4}\right]\cdot$$
$$\cdot\left[1+D+D^2+D^3+D^5+D^7+D^8+D^{11}\right]$$

Thus, the polynomials $g_1(D)=1+D+D^4$ and $g_2(D)=1+D^2+D^3+D^4$ or, in octal form, $(g_1g_2)=(31,27)$. Observe that $g_1(D)$ is primitive so that, for example, $u(D)=1+D^{15}$ produces the $(1+D^{15},1+D+D^2+D^3+D^5+D^7+D^8+D^{11})$ - code, which is finite-length sequence. Of course, any delayed version of this input, say, $D^7.(1+D^{15})$, will simply produce a delayed version of this code sequence.

The Fig. 2 gives one encoder realization for this code. The example serves to demonstrate the conventions generally used in the literature for specifying such encoders.

The function of the permute block is to take each incoming block of N data bits and rearrange them in a pseudo-random fashion prior to encoding by the second encoder.

Unlike the classical interleave block (e.g., block or convolutional interleave block), which rearranges the bits in some systematic fashion.

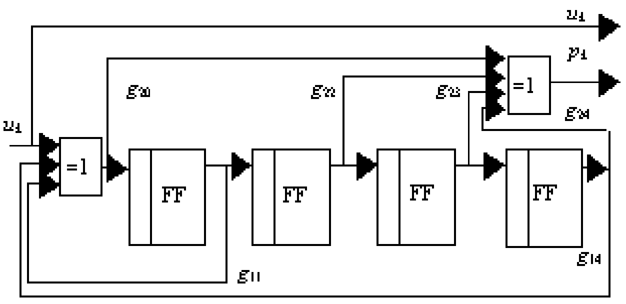

Fig. 2. Recursive encoder $(g_1, g_2) = (31, 27)$

It is important, that the block of interleave sorts the bits in a manner that lacks any apparent order, although it might be tailored in a certain way for weight-two and weight-three inputs.

The indexes of the g coefficients of polynomials express the feedback 1, and the forward 2 directions. The second indexes express the position of coefficient in the polynomial. It is also important that N will be selected quite large and we shall assume N¸ 1000 hereafter.

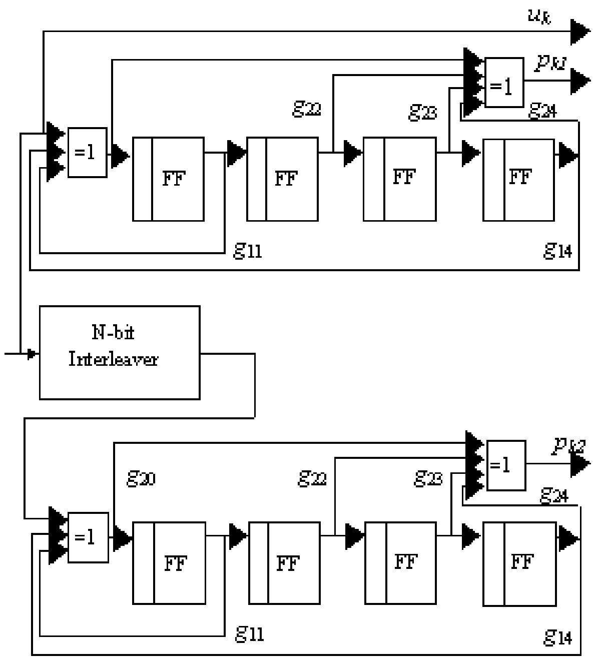

Fig. 3. Turbo code encoder

The importance of these two requirements will be illuminated below. We point out also that one pseudo-random permute block will perform about as well as any other provided N is large. The low-rate code is appropriate, in other situations such as satellite communications, a rate of 1:2 or higher is preferred. The role of the turbo code puncture is identical to that of its convolutional code counterpart, to periodically delete selected bits to reduce coding overhead.

For the case of iterative decoding to be discussed below, it is preferable to delete only parity bits, but there is no guarantee that this will maximize the minimum codeword distance. For example, to achieve a rate of 1:2, one might delete all even parity bits from the top encoder and all odd parity bits from the bottom one.

4. Turbo Decoder

As it will be elaborated upon in the next section, a *maximum-likelihood* (*ML*) sequence decoder would be far too complex for a turbo code due to the presence of the permute block. However, the sub-optimum iterative decoding algorithm to be described there offers near-ML performance. Hence, we shall now estimate the performance of an *ML* decoder (analysis of the iterative decoder is much more difficult).

The idea [5], [6], [8] behind extrinsic information is that *MAP*-D2 provides soft information to *MAP*-D1, using only information not available to *MAP*-D1; the *MAP*-D1 does likewise for *MAP*-D2. The Soft Decision is realized by the à posteriori LLRs (log-likelihood ratios) by the relation

$$\Lambda_i = \ln \frac{P[m_i = 1|y]}{P[m_i = 0|y]}$$

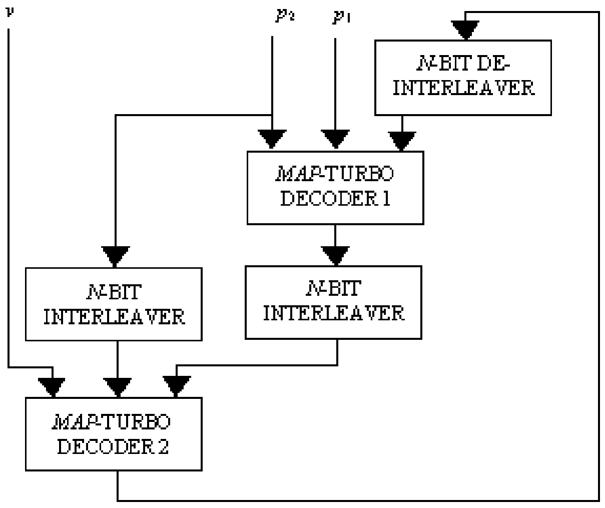

Fig. 4. Iterative MAP-turbo decoder

Decoding algorithm, which processes the a priori information on its input and generates the à posteriori information on its output is called SISO (soft-input, soft-output) decoding algorithm.

The following text will introduce, that the both Viterbi and MAP algorithms can be modified to accept the input data with the soft decision. Similarly, the algorithms can be modified to generate output values with the soft decision. It is clear, from the Figure 5, that the SISO decoder for code rates 1/2 RSC receives three inputs: systematic received sequence $y_i^{(s)}$, of parity symbols $y_i^{(p)}$, and à priori information z_i, which is generated by output of the second decoder. Output variable Λ_i of decoder is defined by previous equation. It is usual to study the systems applying BPSK (binary phase shift keying), which do the modulation by the equation

$$y' = a\sqrt{E_s}(2x-1) + n',$$

where a is an amplitude after feeding, $\sqrt{E_S}(2x\text{-}1)$ is BPSK modulation symbol, $(x \in \{0,1\})$, E_S is an energy for symbol (E_S is in correlation to energy for bit E_b, and to code ratio r, which is defined by relation $E_S = r\,E_b$). A variable n' is the Gauss variable with error $\sigma^2 = N_0/2$. If a is a constant, the information channel is called AWGN (additive white Gaussian noise). In other cases it is called "flat-fading cannel"; the real type of fading depends on statistics a, which is usually described as Rayleigh or Rician. Other statistics can be described as $y=a(2x\text{-}1)+n$, where the error of noise is $\sigma^2 = N_0/2E_s$.

The model of the information cannel, which can be decomposed into three additive components, is used for soft decision LLRs (log-likelihood ratios) on the output of SISO decoder:

$$\Lambda_i = \frac{4a_i^{(s)}E_s}{N_0} y_i^{(s)} + z_i + l_i \,.$$

The variable l_i is called *extrinsic information*. The first two variables on the right-hand side of the equation are in relation to received sequence $y_i^{(s)}$, on the other hand the variable, which is result of the second decoder z_i, the extrinsic information represents a new information, which depends on actual state of decoding process. To prevent problems with the positive feedback, it is important, the extrinsic information be fed from one decoder into the second one. Thus the a priori information on the input of one decoder is calculated by subtracting of the two values of the output – the value is done by expression:

$$l_i = \Lambda_i - \frac{4a_i^{(s)}E_s}{N_0} y_i^{(s)} - z_i \,.$$

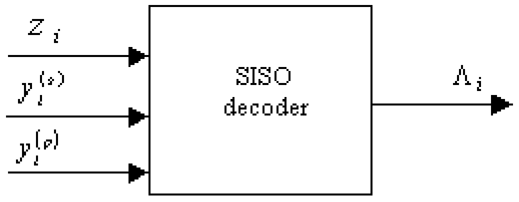

Fig. 5. Decoder with soft decision on the input and with generating of values with soft decision for convolutional code with recurrent generating matrix and ratio $1/2$.

The interconnection of the standard turbo decoder is on the Fig. 6. The first decoder receives the sequence $y^{(0)}$ (multiplied by $4a_i^{(0)}E_s/N_0$), the sequence of control symbols from the first encoder $y^{(1)}$ (multiplied by $4a_i^{(1)}E_s/N_0$) and a priori information $z^{(1)}$, which is calculated by the second one.

5. Turbo Decoder Interconnection

The first decoder generates LLR (log-likelihood ratio) $\Lambda^{(1)}$. The extrinsic information of the first decoder $l^{(1)}$ is yielded as the difference of input variables of the first decoder: a priori received control sequence $r^{(1)}=y^{(1)}.(4a^{(1)}E_s/N_0)$ and the received message $r^{(0)}=y^{(0)}.(4a^{(0)}E_s/N_0)$. The extrinsic information $l^{(1)}$ is used as an a priori information of the second decoder (t.j. $z_{a(i)}^{(2)}=l_i^{(1)}$).

The second decoder receives an interleaved and original message too $\tilde{r}^{(0)} = \tilde{y}^{(0)} \cdot \left(4a^{(0)}E_s/N_0\right)$ and apriori received parity sequence of the message $\tilde{r}^{(2)} = \tilde{y}^{(2)} \cdot \left(4a^{(2)}E_s/N_0\right)$. The second decoder generates LLR (log-likelihood ratio) $\Lambda^{(2)}$, from which extrinsic information $l^{(2)}$ as a result of $\tilde{r}^{(2)}$ subtraction a priori information $z^{(2)}$ is calculated.

This extrinsic information $l^{(2)}$ is re-interleaved and it is used as an input sequence of the first decoder in the next iteration. After Q iterations the final solution of the message is calculated. It is re-interleaved with hard decision by the operation:

$$\hat{m}_i = \begin{cases} 1 & \left(\Lambda^{(2)}_{\alpha(i)} \geq 0\right) \\ 0 & \left(\Lambda^{(2)}_{\alpha(i)} < 0\right) \end{cases}$$

If the code puncturing is used, the decoders do not receive the full parity message sequence. In this case the bits, which were omitted through puncturing, are substituted by zeros. (Linearity of the code makes it possible.)

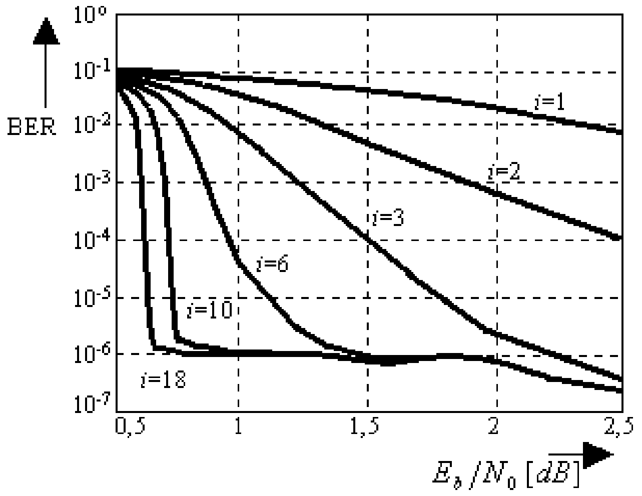

Fig 6. Efficiency of turbo code in dependence on the number of iterations

6. Example

As an illustration of iterative decoding behavior we will consider encoder from Fig 3. If 18 iterations of sequential decoding are calculated, the function on Fig. 7 will be fulfilled.

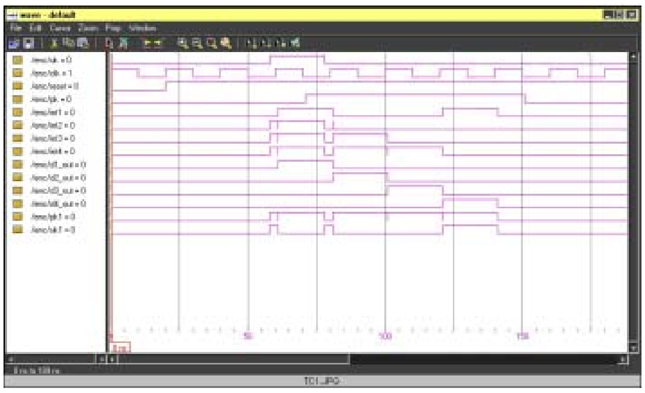

Fig. 7. Simulation of VHDL model 1st step

The variable BER is dramatically enhanced through a bit of iterations. Later the influence becomes weak. This behavior is typical not only for turbo codes, but in other applications, in which iterative decoding is used.

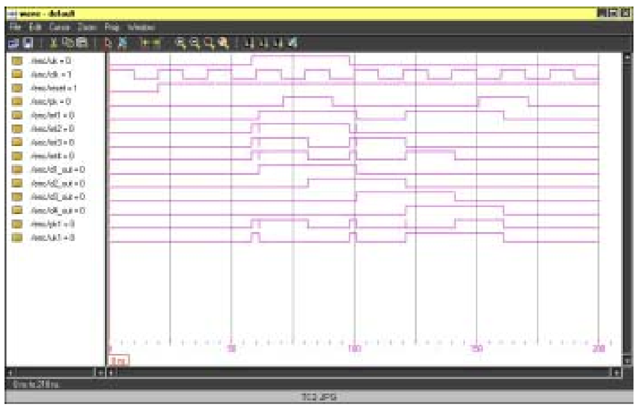

Fig. 8. Simulation of VHDL model 2nd step

7. Conclusion

The discovery of turbo codes, there was much interest in the coding community in sub-optimal decoding strategies for concatenated codes, involving multiple (usually two) decoders operating co-operatively and iteratively.

Most of the focus was on a type of Viterbi decoder [20] which provides soft-output (or reliability) information to a companion soft-output Viterbi decoder for use in a subsequent decoding. Also receiving some attention was the symbol-by-symbol maximum a posteriori (MAP) algorithm.

Acknowledgements

Support for GACR project 102/00/1531 „Formal methods in diagnostics of digital circuits - verification of testable design", is gratefully acknowledged.

References

[1] AITSAB, O., PYNDIAH, R. Performance of Reed-Solomon Block Turbo Code. In IEEE Global Telecomm. Conf. London, UK, Nov. 18-22, 1996, pp. 121-125.

[2] ARNOLD, D., MEYERHANS, G. *The realization of the turbo-coding system.* Semester Project Report, Swiss Fed. Inst. of Tech., Zurich, Switzerland, July 1995.

[3] BAHL, L., COCKE, J.,. JELINEK, F., RAVIV, J. Optimal decoding of linear codes for minimising symbol error rate. *IEEE Trans. Inf. Theory*, Mar. 1974, pp. 284-287.

[4] BENEDETTO, S., MONTORSI, G. Unveiling turbo codes: Some results on parallel concatenated coding schemes. *IEEE Trans. Inf. Theory*, Mar. 1996, pp. 409-428.

[5] BENEDETTO, S., MONTORSI, G. Design of parallel concatenated codes. *IEEE Trans. Comm.*, May 1996, pp. 591-600.

[6] BENEDETTO, S., DIVSALAR, D., MONTORSI, G., POLLARA, F. Serial concatenation of interleaved codes: Performance analysis, design, and iterative decoding. *IEEE Trans. on Inf. Theory,* (March 1997 accepted).

[7] BERROU, C., GLAVIEUX, A., THITIMAJSHIMA, P. Near Shannon limit error- correcting coding and decoding: Turbo codes. In *Proc. 1993 Int. Conf. Comm*., 1993, pp. 1064- 1070.

[8] BERROU, C., COMBELLES, P., PENARD, P., TALIBART, B. IC for turbo-codes encoding and decoding. IEEE Solid-State Circuits Conf., San Francisco, CA, USA, Feb 15-17, 1995, pp. 90-91.

[9] BERROU, C., GLAVIEUX, A. Near optimum error correcting coding and decoding: turbo-codes. *IEEE Trans. Comm.,* Oct. 1996, pp. 1261-1271.

[10] DHOLAKIA, A. *Introduction to convolutional codes with applications*. Kluwer Academic Publishers, 1994, pp. 207-208.

[11] DIVSALAR, D., POLLARA, F. Turbo codes for PCS applications. Proc. 1995 Int. Conf. Comm., 1995, pp. 54-59.

[12] EYUBOGLU, M., FORNEY, G. D., DONG, P., LONG, G. Advanced modulation techniques. Eur. Trans. on Telecom., May 1993, pp. 243-256

[13] HAGENAUER, J., HOEHER, P. A Viterbi algorithm with soft-decision outputs and its applications. Proc. GlobeCom, 1989, pp. 1680-1686.

[14] HAGENAUER, J., OFFER, E., PAPKE L. Iterative decoding of binary block and convolutional codes. *IEEE Trans. Inf. Theory*, Mar. 1996, pp. 429-445.

[15] PEREZ, L., SEGHERS, J., COSTELLO, D. A distance spectrum interpretation of turbo codes. *IEEE Trans. Inf. Theory*, Nov. 1996, pp. 1698-1709.

[16] REED, M., PIETROBON, S. Turbo-code termination schemes and a novel alternative for short frames. IEEE Internat. Symp. on Personal Radio Comm., Taipei, Taiwan, Oct 15-18 1996, pp. 354-358.

[17] ROBERTSON P. Illuminating the structure of code and decoder of parallel concatenated recursive systematic (turbo) codes. Proc. GlobeCom 1994, pp. 1298-1303.

[18] Robertson, P., Villebrun, E., Hoeher, P. A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain. Proc. 1995 Int. Conf. on Comm., 1995, pp. 1009-1013.

[19] UNGERBOECK, G. Channel coding with multilevel/phase signals. *IEEE Trans. Inf. Theory*, Jan. 1982, pp. 55-67.

[20] VITERBI, A. An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes. *IEEE JSAC*, Feb. 1998, pp. 260-264.

[21] VLČEK, K. The VHDL Model of Wyner-Ash Channel Coding for Medical Applications. Proc. of DDECS '98 Workshop, Sept. 2-4, 1998, Szczyrk, Poland, pp. 145-151, ISBN 83-908409-6-0.

[22] VLČEK, K., MIKLÍK, D., KOVALSKÝ, J., MITRYCH, J. Turbo Coding Performance and Implementation. Proc. of the IEEE DDECS 2000 Workshop, 5-7 April 2000, Smolenice, Slovakia, p. 71, ISBN 80-968320-3-4.

[23] VLČEK, K. Turbo-kódy a radiový přenos dat. *Sdělovací technika*, č. 8, 2000, str. 24-26, (In Czech), ISSN 0036-9942.

[24] VLČEK, K. Turbo codes and radio-data transmission. Proc. of the IFAC PDS 2001 Workshop, Elsevier Sci., ltd., Preprints, pp. 22-29.

[25] ZHANG, L., ZHANG, W., BALL, J.T., GILL, M.C. Extremely robust turbo coded HF modem. Proc. of Conf. MILCOM 96, Oct 21-24, 1996, Washington, DC, USA, pp. 691-695.

About Author...

Karel VLČEK was born in Zlin, Czech Republic. He received his M.Sc. (Ing.) degree at the University of Technology in Brno, his PhD. (CSc.), and Assoc. Prof. (Doc.) at the Czech Technical University in Prague. He interests in Computer Science, Diagnostics, Design Automation, Error-Control Coding, Signal Processing, Multimedia and Biomedicine. He was solver of the international (EU) and Czech research projects, GAČR, FRVŠ agency of Ministery of Education and Sports.