A Generalization of the Convolutional Codes

Dejan Spasov
Faculty of Computer Science and Engineering
Sts. Cyrill and Methodius University
Skopje, North Macedonia

***Abstract*—In this paper we propose a generalization of the convolutional codes. The proposed generalization of the convolutional codes offers the possibility to discover previously unknow convolutional codes. For example, convolutional codes with pseudo-random time-varying trellis diagram that may improve error-correcting capabilities of the convolutional codes. An important property of the proposed generalization of the convolutional codes is that the decoding complexity remains the same as the decoding complexity of the ordinary convolutional codes. We propose encoding and decoding schemes for the generalized convolutional codes.**

Keywords—error-correcting codes, convolutional codes, turbo codes, LTE, 5G

I. INTRODUCTION

Channel coding plays an important part in modern communication systems. In [1], Shannon gave probabilistic proof that we can communicate with an arbitrary small probability of error as long as the communication rate is below the channel capacity. Error-correcting codes provide constructive solution to the Shannon's theorem. Many error-correcting codes were developed to provide low-power and reliable communication over unreliable channels.

Let F be a finite alphabet of $|F|$ letters and let F^n be the set of all strings of length n over F. In general, an *error-correcting code* C is a subset of F^n of M elements. Elements of the code $c_i \in C$ are called *codewords.*

Let $d(x, y)$ denote the *Hamming distance* between two strings $x, y \in F^n$. The *Hamming distance* $d(x, y)$ is the number of positions in which x and y differ. Let d denote the *minimum distance* d of the code C defined as

$$d = \min\{d(c_i, c_j) | c_i, c_j \in C, i \neq j\} \quad (1)$$

If the minimum distance d is known, we say that C is an (n, M, d) code. The larger the minimum distance d of a code C is, the better the error-correcting capability the code C is.

The code C is linear if its codewords form k- dimensional linear subspace in F^n. We will write $[n, k, d]$ to denote that the code C is linear. For linear codes there exist k basis vectors that are kept as rows in a *generator matrix* G. For each linear code C there is a generator matrix G of type $G = [I\ A]$, where I is the identity matrix. We may say that the generator matrix $G = [I\ A]$ is in *standard form*. It is well-known that for linear codes there exist additional matrix, known as the *parity check matrix* H, defined as

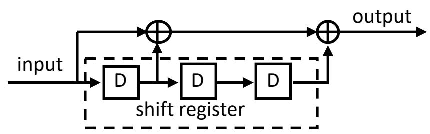

Fig. 1. A convolutional encoder

$$Hc_i^T = 0, \forall c_i \in C \quad (2)$$

Let $G = [I\ A]$ be the generator matrix of a code C, then $H = [-A^T\ I]$is the parity check matrix of the code C.

A k-letter message may be encoded in a n-letter codeword c. The codeword c may be sent over a noisy channel. The noisy channel may alter few bits of the codeword. A receiver may receive a n-bit string x obtained from the codeword c in which some bits have been altered. The process of finding the nearest, in terms of Hamming distance $d(x, y)$, codeword $\hat{c} \in C$ to the string x, is known as *decoding*

$$\hat{c} = \operatorname{argmin}\{d(x, c_j) | c_j \in C\} \quad (3)$$

Some codes have efficient (polynomial) procedure to find the nearest codeword $\hat{c} \in C$ (in terms of Hamming distance) to the received string x. Codes with polynomial decoding procedures can be used for transmission of digital information over a noisy channel.

In general, error-correcting codes may be divided in two groups: block codes and convolutional codes [2], [3]. Block codes offer greater error-correcting capability, but their decoding algorithms are not very efficient. Convolutional codes are a class of error-correcting codes with polynomial encoding and decoding procedures. They are used in numerus applications to achieve reliable data transfer and reliable data storage. For example, convolutional codes are used in digital video storage, satellite communications, GSM networks, numerous standards: GPRS, EDGE, LTE, 3G, and so on. In this paper we propose a framework that generalizes the concept of convolutional codes. In Section II we briefly introduce certain aspects of convolutional codes that are important for presenting our idea. In Section III we present the new class of convolutional codes. We consider these codes to be generalization of the convolutional codes. We propose

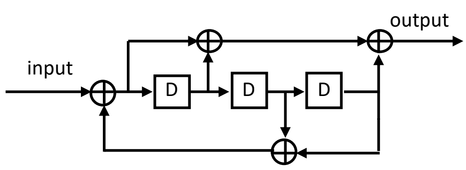

Fig. 2. A recursive convolutional encoder

Funded by a project from the Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University.

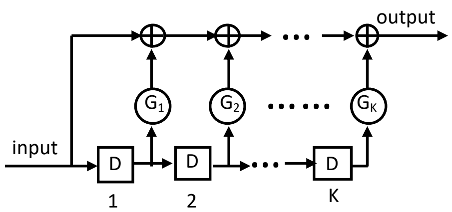

Fig. 3. A framework for a convolutional encoder with K-bit shift register

encoding and decoding mechanisms. In section IV we discuss possible applications of the new class of convolutional codes.

II. Convolutional Codes

Convolutional codes are one of the oldest known classes of error-correcting codes. They were introduced by P. Elias in 1955 [2]. Fig. 1 shows a circuitry that may be used for producing convolutional codes. This circuitry is known as *convolutional encoder*. The convolutional encoder is built with a shift register of length 3 and combinatorial logic elements, i.e. XOR gates. The shift register consists of 3 memory cells (denoted with D) that introduce one-bit delay. In one clock cycle, one information bit enters the shift register from the left side. Data bits that are already in the register are shifted to the right. Let a semi-infinite stream of information bits $x = (x_1, x_2, \dots, x_i, \dots)$ is provided to the input of the convolutional encoder. In this sequence an information bit x_i is considered older than the information bit x_{i+1}. Let the convolutional encoder be configured to output a semi-infinite sequence of *parity bits* $y = (y_1, y_2, \dots, y_i, \dots)$. The principle of operation of the convolutional encoder is simple: at time $t = 0$ the content of the shift register is 0; a parity bit y_i is computed as

$$y_i = x_i \oplus x_{i-1} \oplus x_{i-3}. \tag{4}$$

where, the symbol $\oplus$ denotes the binary operation XOR.

Convolutional encoders may be divided in two major categories: *recursive* convolutional encoders and *non-recursive* convolutional encoders. The encoder on fig. 1 is an example of a non-recursive encoders. Fig. 2 shows an example of a recursive convolutional encoder. A recursive convolutional encoder is obtained from a non-recursive encoder with two branches of combinatorial elements that may output simultaneously two parity bits. One of the two

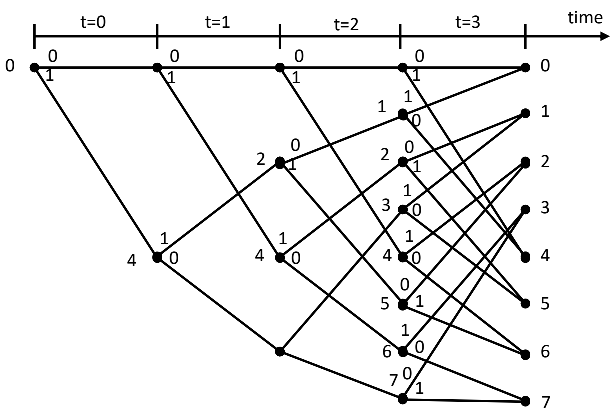

Fig. 4. Trellis diagram of a convolutional encoder

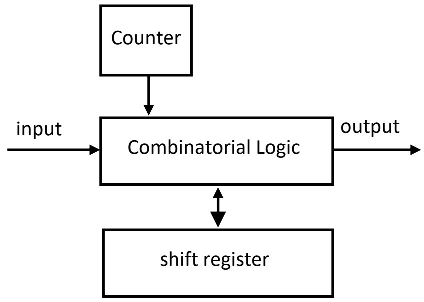

Fig. 5. Proposed convolutional encoder

branches is fed back to the input of the shift register. This way, in a recursive convolutional encoder outputted parity bit is influenced by all older information bits.

Fig. 3 shows a framework for a convolutional encoder with K-bit shift register. The K-bit shift register consists of K memory cells (denoted with D) that introduce one-bit delay. We can obtain a particular encoder using the framework by specifying $G_i = 1$ or $G_i = 0$ for all elements G_i, $i = 1, 2, \dots, K$. Here $G_i = 1$ denotes short circuit and $G_i = 0$ denotes high impedance. An encoder is completely defined by specifying all elements G_i. Particular values of the elements G_i may be represented with a *generator vector* $G = [G_1, G_2, \dots, G_K]$. Sometimes, the generator vector G may be given as octal number. For example, the generator vector for the encoder on fig. 1 is $G = 5$. Hence, outputted bits of a convolutional encoder with shift register of length K are computed as

$$y_i = x_i \oplus G_1 \cdot x_{i-1} \oplus \cdots \oplus G_K x_{i-K} \tag{5}$$

The length K of the shift register is known as *constrained length* of the convolutional code. In general, larger shift register provides larger number of input bits that influence outputted parity bits. It is believed that larger constrained length K may provide a convolutional code that may recover larger number of errors. On the other side, larger constrained length K makes the decoding of the convolutional code more complex and demanding in terms of time, space, and hardware.

A convolutional encoder with K shift registers can be considered a finite state machine with 2^K states. A state in the finite state machine is identified with the content of the shift register. Transition from one state to another produces a parity

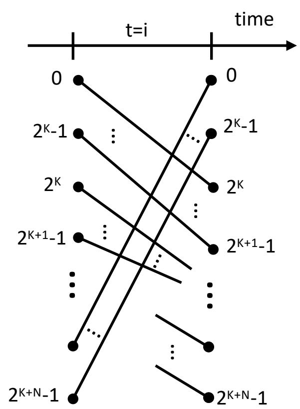

Fig. 6. Trellis diagram of the proposed convolutional encoder

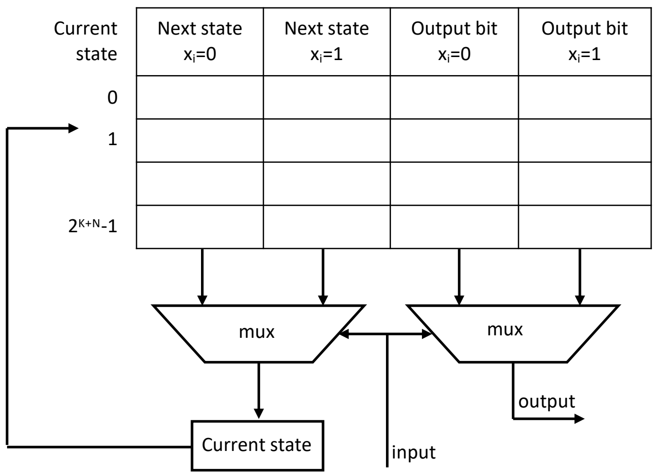

Fig. 7. Implementation of the proposed convolutional encoder

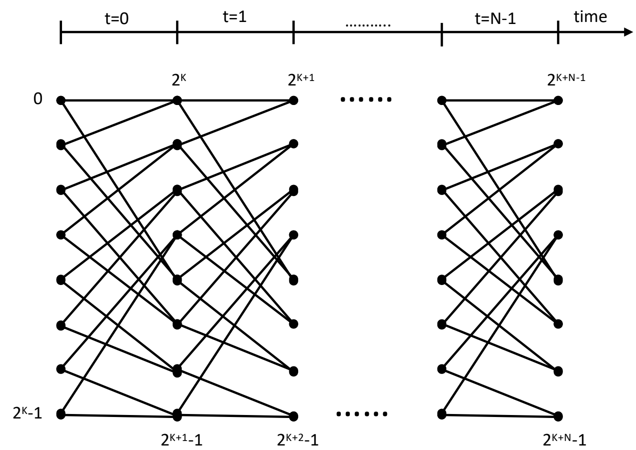

Fig. 8. Decoding of the proposed convolutional encoder

bit. Given a finite state machine, progression of the machine through the states with respect to the time is known as *trellis diagram* of the machine. Trellis diagram is labelled *M*-partite graph, in which every path represents a valid codeword. Fig. 4 shows the trellis diagram that corresponds to the convolutional encoder given on fig. 1. Vertices are labeled with decimal numbers from 0 to 7. Vertices represent all states of the shift register. Edge labels represent parity bit that will be outputted for that transition.

Encoding process of both the block codes and the convolutional codes may be described with trellis diagram. Trellis diagram of error-correcting codes give hint about the decoding. A decoder may be configured to guess the mos likely path through the trellis. The block codes, in general, have exponential number of states; thus, the decoding process is much complex. The number of states of the trellis diagram of a convolutional code is determined by the size of the shift register. Hence, convolutional codes have better decoding complexity; although, convolutional codes have weaker error correcting capability. In general, all convolutional codes can be decoded with the Viterbi algorithm [5], [6]. Memory requirements of the Viterbi decoder are proportional with the constrained length K. Another algorithm that may be used for decoding is the BCJR algorithm [7]. Memory requirements of the BCJR decoder are proportional with the product of the length of the sequence and the number of states of the shift register.

Popular convolutional codes are the turbo codes. Turbo codes were the first error-correcting codes that demonstrated reliable communications near the channel capacity with practically feasible hardware [8]. Due to their excellent error-correcting capability, they are part of many modern communication technologies, like LTE [9]. An LTE turbo encoder is a systematic encoder made of two 8-state recursive convolutional encoders. Recursive encoder used in turbo codes is given on fig. 2. Generator polynomials of the LTE turbo codes are fixed and specified in a standardized specification [9].

III. Generalized Convolutional Codes

From fig. 3 it can be observed that the framework with K-bit shift register can specify 2^K convolutional encoders. Thus, only 2^K non-recursive encoders can be built. Additionally, this framework specifies that only 2^{K+1} recursive convolutional encoders can be built. However, the number of possible convolutional encoders is much larger than 2^K. In this paper, our goal is to provide a framework for a convolutional encoder that may specify all possible convolutional encoders. Principle of operation of the proposed framework is given on fig. 5.

The framework on fig. 5 generalizes the concept of convolutional encoder. The convolutional encoder consists of a K-bit shift register, a combinatorial logic, and a N-bit shift register. Principle of operation of the shift register and the combinatorial logic is similar to the principle of operation of the regular convolutional encoder (fig. 1 and fig. 2). The combinatorial logic convolutes the bits stored in the shift register and the bit provided to the input in order to produce an output bit. A counter is added to the framework. The counter is incremented with each input bit. In a simplest embodiment, the counter may be 1-bit counter. The purpose of the counter is to increase the number of states of the encoder. A state may be identified with the content of the counter and the content of the shift register. Thus, the proposed encoder has 2^{K+N} states. Each time the counter is incremented, the combinatorial logic changes the formula for computing the output bit. Hence, the parameters of the encoder appear to vary over time.

The proposed encoder may be considered as finite state machine with 2^{K+N} states. The trellis diagram of the finite state machine is given on fig. 6. It can be observed that the trellis diagram has larger number of states compared to the trellis of a regular convolutional encoder (fig. 4). An important feature of the trellis is that the 2^{K+N} states may be divided into 2^N disjoint sets, wherein each set comprises of 2^K states. Transitions are possible from one set of states to another set. This property makes the decoding of the proposed convolutional codes is with the same complexity as the regular convolutional codes.

Hardware implementation of the proposed convolutional encoder consists of a SRAM memory operated as table, a register to hold the current state and multiplexers (fig. 7). The table is indexed with the number of states of the finite state machine. The content of the current-state register is provided as index to the read ports of the table. The output of the table is provided as input to the multiplexers. An input bit is provided as selection control to the multiplexers. This way, the appropriate fields of the outputted entry are selected.

The proposed convolutional codes can be decoded with the Viterbi algorithm and with the BCJR algorithm. Decoding complexity of the proposed codes remains the same as decoding complexity of the regular convolutional codes. Fig. 8 shows another view of the trellis diagram of the proposed codes. The diagram shows only possible transitions from one set of states to another set of states at one point of time. It can be observed that at any time, the number of active states is the same as with the regular convolutional codes. Therefore, the decoding complexity remains the same.

IV. CONCLUSION

We have proposed a framework for generating a new class of convolutional codes. We consider the new class of codes as generalization of the concept of a convolutional code. The proposed framework offers possibility to specify convolutional codes that cannot be specified with the previously known methods for specifying convolutional codes. One important feature of the proposed convolutional codes is that the decoding complexity remains the same as the decoding complexity of the regular convolutional codes, although the number of states increases. The proposed encoder can be efficiently implemented in hardware using SRAM memory cells operated as table. The SRAM table may be populated with the possible transitions at the run-time. Therefore, the parameters of the convolutional codes need not to be specified at the design stage of the communication system. Our future goal is to search over theses codes and to find a convolutional code with improved error correcting capability.

REFERENCES

[1] C. E. Shannon, "The mathematical theory of communication," Bell System Technical Journal, vol. 27, pp. 379–423, July 1948.

[2] F. J. MacWilliams, N.J.A. Sloane, "The Theory of Error-Correcting Codes," North Holland, Amsterdam, 1977.

[3] S. Lin, D. J. Costello, "Error Control Coding," Pearson, 2005.

[4] Elias, P. "Coding for noisy Channels." *IRE Convention Record*, Part IV, pp. 37-46 (1955).

[5] Viterbi, A. J. "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm." *IEEE Transactions on Information Theory* vol. 13, no. 2, pp. 260–269, 1967.

[6] Forney, G. D. "The Viterbi Algorithm." *IEEE Transactions on Information Theory* vol. 61, no. 3, pp. 268–278, 1973.

[7] Bahl, L., Cocke, J., Jelinek, F., Raviv, J. "Optimal Decoding of Linear Codes for minimizing symbol error rate." *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp.284-287, 1974.

[8] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in Proc. ICC, Geneva, Switzerland, May 1993.

[9] ETSI Technical Specification, "LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and channel coding," 3GPP TS 36.212 version 8.8.0 Release 8, Jan. 2011.