

# Parallel Concatenated Codes for Iterative Source-Channel Decoding

Ksenija Laković and John D. Villasenor

University of California, Los Angeles

Electrical Engineering Department

Los Angeles, CA 90095-1594

{ksenija, villa}@icsl.ucla.edu

## 1 Introduction

This paper considers iterative decoding of reversible variable length codes (RVLC) in conjunction with convolutional channel codes (CC). As the major reference we use the recently proposed scheme [1], based on serially concatenated RVLC and CC soft-output decoders. This scheme utilizes simple constituent modules and significantly outperforms a system in which source and channel decoding are performed in a non-iterative manner.

We propose an alternative coding scheme, with the encoder that involves one simple extra element, and show how to construct the iterative decoder based on parallelly concatenated source and channel codes. Relative to [1], the proposed system exhibits decoding convergence at a significantly lower  $E_b/N_o$ .

## 2 Serially and Parallelly Concatenated Decoders

Consider a transmitter of a communication system illustrated in Figure 1(a), which consists of a RVLC encoder, an interleaver and a convolutional encoder. This structure is common in practice, and has been studied in [1]. A suitable decoder for this system can be constructed by a serial concatenation of source and channel codes, as illustrated in Figure 1(b). The decoding algorithm is based on well-know iterative decoding principles and we refer the reader to [1,2] for more details. The inputs and outputs of a SISO module are indicated in Figure 1(c), and are similar to the notations in [2].

We propose the encoder shown in Figure 1(d), and assume that the recursive convolutional code unit (RCC) is the same as in Figure 1(a). There are several ways to construct a decoder for this system, and we propose the structure shown in Figure 1(e). It consists of two parallelly concatenated SISO modules. One of them is associated with the recursive convolutional code (RCC), while the other is associated with both the RVLC and the accumulator. Essentially, we design a trellis for joint RVLC-ACC decoding and incorporate it in the decoding process, instead of separate ACC and RVLC trellises.

It may appear that the proposed scheme has disadvantages related to the increased decoding complexity. In general, a concatenation of a  $V$ -state trellis and a  $W$ -state trellis yields a  $(V \cdot W)$ -trellis; therefore the number of states of the RVLC-ACC trellis is two times higher than of the RVLC trellis (ACC trellis is a 2-state trellis). This effect can be compensated, however, by use of a lower-complexity RCC code. In the following section we will show that the proposed system exhibits good performance utilizing simple RCC codes and outperforms serially concatenated schemes of similar and higher overall complexity.

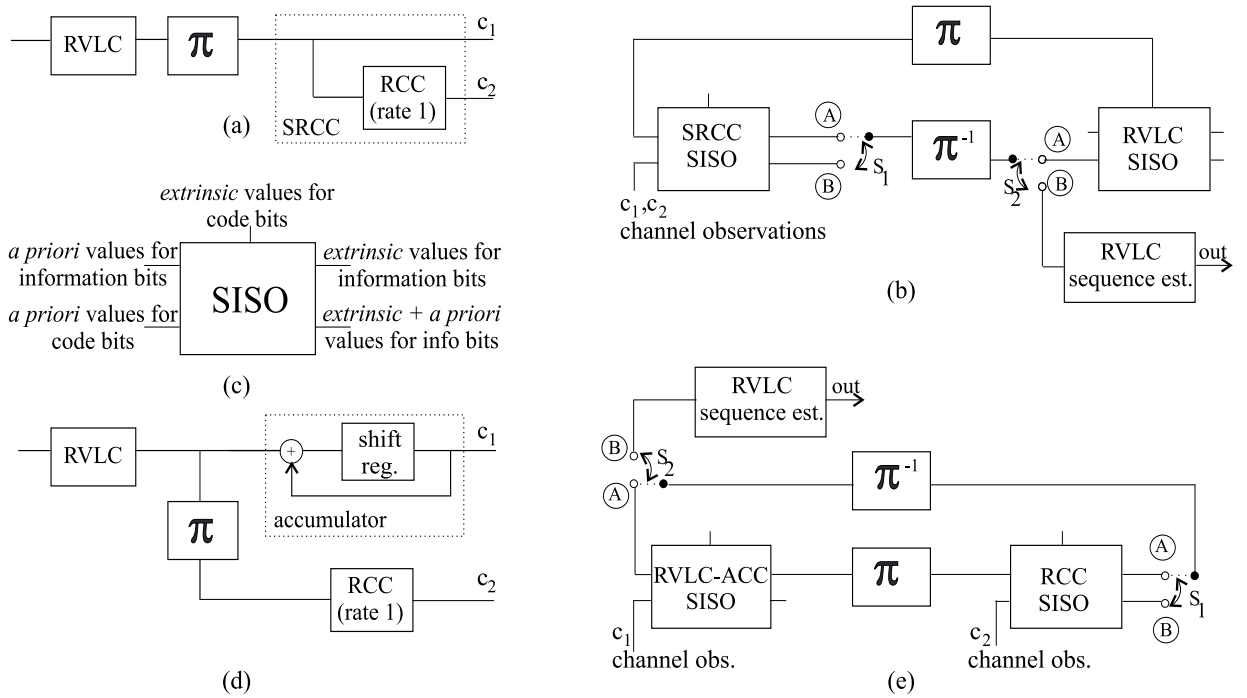


Figure 1: (a) Encoder as in [1], with a rate 1/2 systematic recursive convolutional code (SRCC). (b) Serially concatenated (SC) decoding scheme, similar to [1]. (We have added switch  $S_1$  to improve performance.) The switches are in position A during the iterative decoding process and in position B at the last step. (c) Inputs and outputs of SISO modules. (d) Proposed encoder. (e) Parallely concatenated (PC) decoding scheme.

### 3 Results

Simulations were performed for the source as in [1]. In contrast with [1], the data stream was not divided into blocks containing a fixed number of symbols ( $K$ ). Instead, we used blocks containing a specified number of bits ( $N = 2000$ ), allowing a few-bit extensions if necessary to complete the bit-stream associated with the symbol at the end of a block. We prefer this approach for two reasons. The first one is that the side information on the block-length, which should be reliably transmitted to the decoder, is reduced from the order of the frame size ( $N \sim 10^3$ ) to the order of the longest RVLC codeword ( $\sim 10$ ). Another reason is that the restricted variation in length simplifies the problem of variable-length sequence interleaving.

We first compare the parallely concatenated (PC) scheme to the serially concatenated (SC) scheme, assuming that the same RCC is used in both cases. In Figure 3(a) we present simulation results for both schemes, using an 8-state RCC code with the octal generator 17/13. The symbol error rate (denoted as  $SE_R^L$ ) is evaluated in terms of the Levenshtein distance [3], as in [1]. As argued in [4] this metric accounts for the self-synchronizing properties of variable length codes, and thus is more suitable than a metric based on a simple symbol comparison.

Results presented in Figure 3(a) demonstrate that the PC scheme, with 4 decoding iterations, for symbol error rates  $10^{-4}$  and lower, outperforms the SC scheme by more than 0.5dB. With the increased number of iterations the difference between the schemes is even bigger; for 12 iterations and low symbol error rates the PC scheme outperforms the SC scheme by about 0.8dB. Note that in this case the SC scheme is about 28% less complex than the PC scheme. The complexity is evaluated by the total number of trellis edges associated with each decoding step, which is shown in Figure 2.

In Figure 3(b) we compare the same PC scheme against several SC schemes with different RCC units. Results demonstrate that the PC scheme outperforms SC schemes of similar and higher complexity.

scheme	RCC octal generator	number of trellis edges					relative complexity vs. PC scheme
		RCC	SRCC	RVLC	RVLC-ACC	total	
PC	17/13	16	/	/	20	36	1.00
SC <sub>1</sub>	17/13	/	16	10	/	26	0.72
SC <sub>2</sub>	33/23	/	32	10	/	42	1.17
SC <sub>3</sub>	67/45	/	64	10	/	74	2.05

Figure 2: Simulated coding schemes. RCC coding generators are taken from [5]. Relative complexity of a scheme is computed as (total edges of the scheme)/(total edges of PC).

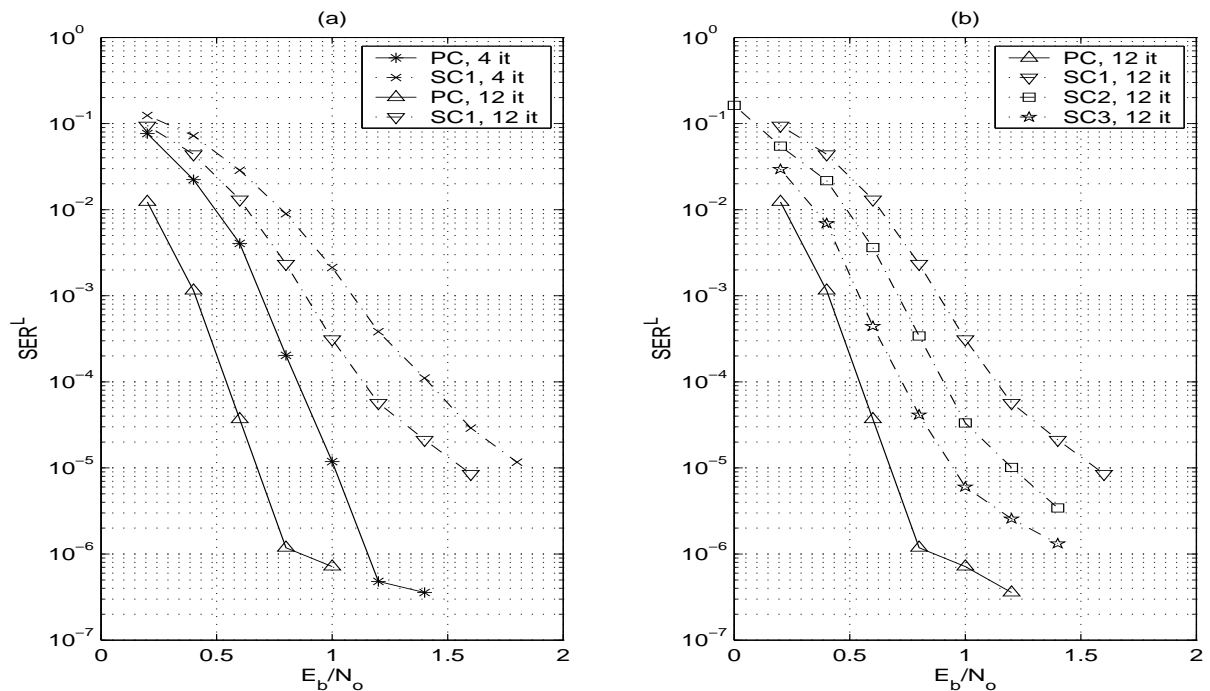


Figure 3: (a) PC scheme compared to SC scheme with the same convolutional code. (b) PC scheme compared to SC schemes of different complexity.

## References

- [1] R. Bauer, J. Hagenauer, "On variable length codes for iterative source/channel-decoding," *Proceedings Data Compression Conference*, Snowbird, UT, USA, 2001.
- [2] J. Hagenauer, "The turbo principle - tutorial introduction and state of the art," *Proc. of the Int. Symp. on Turbo Codes and Related Topics*, Brest, France, 1997.
- [3] T. Okuda, E. Tanaka, T. Kasai, "A method for the correction of garbled words based on the Levenshtein metric," *IEEE Transactions on Computers*, vol. C-25 (no.2), 1976.
- [4] R. Bauer, J. Hagenauer, "Iterative source/channel-decoding using variable length codes," *Proceedings Data Compression Conference*, Snowbird, UT, USA, 2000.
- [5] M. S. C. Ho, S. S. Pietrobon, T. Giles, "Improving the constituent codes of turbo encoders," *IEEE GLOBECOM 1998*, Sydney, Australia, 1998.