Bridging the Gap
Between Parallel and Serial Concatenated Codes

Naveen Chandran and Matthew C. Valenti
Wireless Communications Research Laboratory
West Virginia University
Morgantown, WV 26506-6109, USA
email: chandran@csee.wvu.edu, mvalenti@wvu.edu

Abstract

Previously [1], it has been shown that parallel concatenated convolutional codes (PCCCs) can be modeled as a special case of serial concatenated convolutional codes (SCCCs). In this paper, we focus on this relationship with a goal of providing a parent code design that generates PCCC, SCCC and a family of hybrid code performances that bridge the gap between the two. The proposed code is very flexible since a single encoder can produce the entire range of outputs while possessing the same decoder structure to retrieve the input data. Simulation results of the error rate performance of these codes vs. signal to noise ratio are plotted. Finally, an insight into design and analysis of good parent codes is provided.

1 Introduction

Turbo codes are typically classified into two broad categories: Parallel concatenated convolutional codes (PCCC) and serial concatenated convolutional codes (SCCC). PCCCs perform exceptionally well at low signal-to-noise ratios (SNRs) but develop rather high error floors at high SNRs [2]. On the other hand, SCCCs can achieve extremely low bit error rates at high SNRs, although this comes at the cost of worse performance (relative to PCCCs) at very low SNRs [3].

Until recently, system designers who wish to employ turbo codes have had to decide between using PCCCs (with their superior low SNR performance) and SCCCs (with the greatly reduced error floor). However, Divsalar and Pollara have shown the possibility of hybrid concatenated codes, which combine serial and parallel code concatenations in such a way that satisfactory performance is achieved in all SNR regions [4]. Furthermore, in a paper by Wu and the second author, it was shown that there is a close relationship between PCCCs and SCCCs, and that, in fact, PCCCs are a special case of SCCCs [1]. In particular, a PCCC can be created from a SCCC as long as the following three conditions are satisfied: (1) Both the inner and outer encoders are RSC encoders, (2) The SCCC interleaver is designed to output all of the systematic bits from the outer encoder before it outputs any of its parity bits, and (3) All of the so-called "double-parity bits" (the parity output of the inner encoder generated using the parity output from the outer encoder) are punctured. Therefore, it is the puncturing of the double-parity bits that separates the PCCC from its SCCC cousin.

By noting the close relationship between PCCCs and SCCCs, three interesting observations can be made. First, since it is possible to encode a PCCC using a SCCC encoder, it is likewise possible to decode a PCCC using a SCCC decoder. This implies that a SCCC codec is more flexible than a PCCC codec, and that perhaps IC manufacturers should focus their efforts on SCCC products. Second, an interesting ARQ/FEC scheme with incremental redundancy technique is possible whereby at first, only those bits making up the PCCC code are transmitted, while the additional bits that turn the PCCC code into a SCCC code are sent later, but only if requested (in the form of a negative-acknowledgement) [1]. The third observation, and the focus of this paper, is that by only deleting some of the double-parity bits, rather than all (for the PCCC case) or none (for the SCCC case) of them, it is possible to generate hybrid turbo codes whose performance bridges the gap between the PCCC and SCCC cases. Thus the decision to use PCCC or SCCC codes no longer needs to be "black or white", rather a middle ground (shades of "gray") exist that can give the system designer more flexibility.

Note that our interpretation of hybrid codes is quite different than that in [4]. In [4], two encoders in a serial concatenated structure form a parallel combination with a third convolutional encoder. Three encoders and two interleavers (one for each concatenation) were used in a structure called hybrid concatenated convolutional code (HCCC). Our system only requires two encoders and a single interleaver.

In this paper, we propose a parent code model to generate PCCC, SCCC and hybrid turbo codes that bridge the gap between the two. In the following sections, first, we present the system model of the parent code. Second, we discuss the implementation issues of the proposed model. Third, we provide BER vs. SNR simulation results of the model for a number of different frame sizes. Fourth and finally, we leverage density evolution analysis from [5] and iterative decoding convergence analysis from [6] to assist the design of good parent codes.

2 System Model

We use the system model in Fig. 1 to develop the proposed parent code. The system model consists of a SCCC encoder, whose output is punctured based on an appropriate puncturing

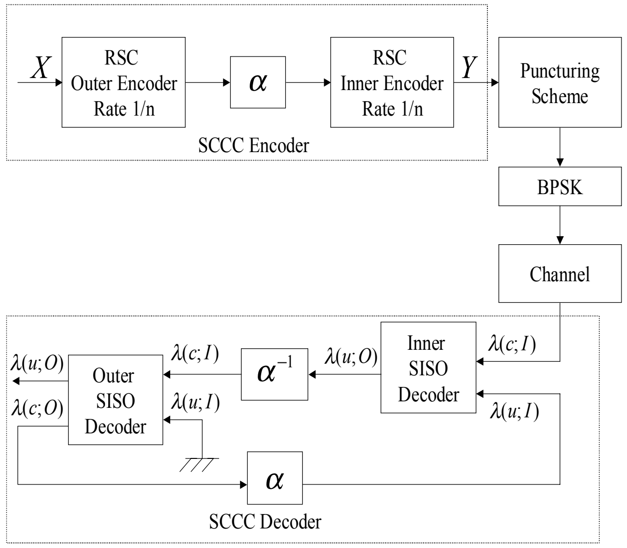


Figure 1: System model of a parent SCCC code.

scheme. The punctured encoder output is passed through an AWGN channel after BPSK modulation. The channel output is fed to an iterative SCCC soft-input soft-output (SISO) decoder, which estimates the data and code bits.

The SCCC encoder comprises of two rate $\frac{1}{2}$ recursive systematic convolutional (RSC) encoders. The information bits are first encoded by the outer $r = \frac{1}{2}$ RSC encoder. The systematic and parity bits generated by the outer encoder are fed to a spread interleaver (α) [7]. In our model, however, the spread interleaver is structured such that it outputs all the systematic bits appearing at the output of the outer encoder before it outputs the encoder's parity bits. The interleaved bits are then fed to the inner RSC encoder. The interleaver structure helps to identify the information at the output of the inner encoder before a selective puncturing scheme is applied. The structuring of the interleaver and the puncturing schemes used to generate PCCC and hybrid turbo codes from a transformed SCCC parent code are discussed further in Section 3. The trellis of both the inner and outer RSC encoders are terminated.

Apart from the interleaving (α) and deinterleaving (α^{-1}) patterns, the SISO SCCC decoder that we use is identical to the conventional SCCC decoder. In Fig. 1, the channel values ($\lambda(c; I)$) are fed to the inner decoder whose a priori information ($\lambda(u; I)$) is initially set to zero. Extrinsic information messages ($\lambda(u; O)$ and $\lambda(c; O)$) in terms of log-likelihood ratios (LLRs) are passed back and forth between constituent SISO decoders during iterations. After the final iteration, the data bits are estimated based on the LLRs of the information bits ($\lambda(u; O)$) output by the outer decoder.

3 Interleaver Structuring and Parent Code Design

As discussed in the previous section, the structured spread interleaver outputs all the systematic bits before any parity bits are output. In other words, the interleaved frame (frame at the output of the interleaver) is broken down into two halves and the interleaver maps the systematic bits to the first half of the interleaved frame while the parity bits are mapped to the second half. Since the systematic and parity bits are multiplexed to form the output of the RSC encoder, it can be said that the interleaver must

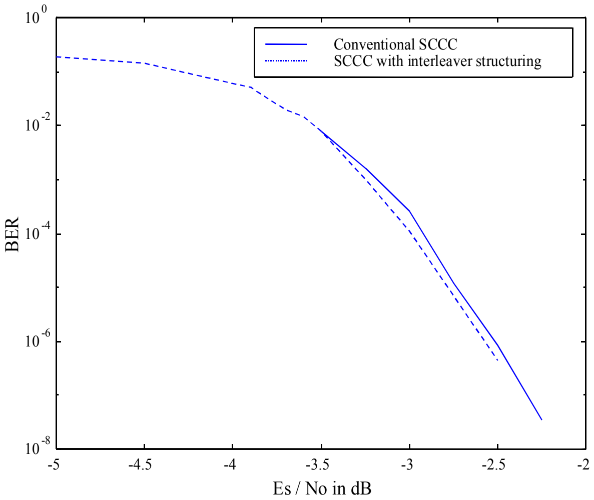
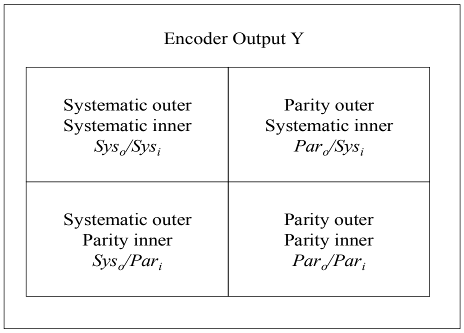


Figure 2: Performance comparison of a $r = \frac{1}{3}$ SCCC code with and without interleaver structuring.

map the odd ordered bits[1] (systematic bits) to the first half and the even ordered bits (parity bits) to the second half of the interleaved frame.

Once the process of structuring is completed, generating a spread interleaver (with a spreading factor S) is only a matter of checking S positions on either side of the half way mark of the interleaved frame for the presence of adjacent bits of the encoder ouput. Moreover, the quality of the interleaved frame (average of all the distances in the interleaved frame between adjacent bits of the original frame) generated with this interleaver structuring is exactly the same if not better than the conventional spread interleaver generated frame. This is revealed by the result in Fig. 2, which is a comparison plot of the SCCC performance with and without incorporating the interleaver structure.


Encoder Output Y

Systematic outer Systematic inner Sys_o/Sys_i	Parity outer Systematic inner Par_o/Sys_i
Systematic outer Parity inner Sys_o/Par_i	Parity outer Parity inner Par_o/Par_i

Figure 3: Output information from SCCC encoder after structured interleaving.

The SCCC codes used to simulate the curves in Fig. 2 are composed of a pair of identical constraint length K = 5, generator polynomial [35,23] (in octal) RSC codes with linear log-MAP SISO constituent decoders [8]. These codes are punctured to a rate $r = \frac{1}{3}$ and passed over an AWGN channel. The simulation parameters mentioned above are used in all the simulations throughout this study.

The rate $r = \frac{1}{3}$ codes are generated from rate $r = \frac{1}{4}$ SCCC codes by puncturing every alternate parity bit at the output of the inner encoder. A frame size of 512 data bits was used for both codes. The solid line in the figure is the conventional SCCC performance while the dotted line is the performance of SCCC with the interleaver structure.

3.1 PCCC from an SCCC codec

The interleaver design permits the output bits of the inner encoder to be categorized into four fields[2] as shown in Fig. 3: Sys_o/Sys_i, Par_o/Sys_i, Sys_o/Par_i and Par_o/Par_i. When

[1] assuming the indexing starts from 1.

[2] Sys and Par for systematic and parity respectively, and o and i for outer and inner encoder respectively.

compared with the output information from a PCCC, it can be seen that the Sys_o/Sys_i, Par_o/Sys_i, and Sys_o/Par_i fields of the SCCC are equivalent to the systematic information from RSC1, parity information from RSC1, and parity information from RSC2 of the PCCC respectively [1].

Thus, only the Par_o/Par_i (also called double-parity) bits differentiate a PCCC from a SCCC. This is confirmed by the result in Fig. 4, which is a performance comparison plot of a conventional $r = \frac{1}{3}$ PCCC and a $r = \frac{1}{3}$ PCCC generated from the SCCC codec by puncturing all the double parity bits. The simulation parameters used for the PCCC comparison in Fig. 2 are exactly the same as those used in the SCCC comparison of Fig. 2: K = 5, g = [35,23], rate $r = \frac{1}{3}$, frame size 512 and an AWGN channel.

In Fig. 4, the solid line represents the conventional unpunctured rate $r = \frac{1}{3}$ PCCC performance while the dotted line shows the performance of a PCCC generated from an equivalent $r = \frac{1}{3}$ SCCC codec. It is seen that the double-parity punctured SCCC performs exactly the same as conventional PCCC.

3.2 Hybrid Turbo Codes

It has been established that a PCCC can be generated from a SCCC by puncturing all the double-parity bits. This concept gives rise to the fact that multiple new hybrid code designs can be constructed from a parent SCCC code by varying the number of double-parity bits punctured and puncturing the remaining bits from the single-parity fields to maintain the same overall code rate.

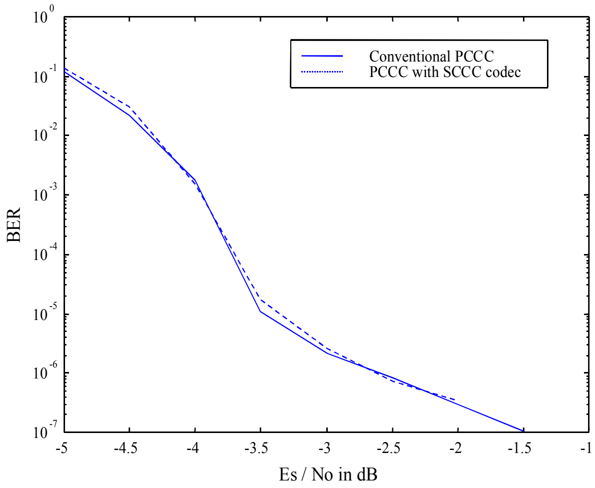


Figure 4: Performance comparison of a conventional $r = \frac{1}{3}$ PCCC and a PCCC generated by puncturing all the double parity bits of an equivalent $r = \frac{1}{3}$ SCCC codec.

Hybrid codes can be designed to combine the advantages of the PCCC and the SCCC, thereby, bridging the gap between the two. Hybrid codes with a large number of punctured double-parity bits will possess more PCCC-like properties and perform better than SCCC in the low SNR region while those codes whose double-parity bits are punctured less heavily will possess more SCCC-like properties and have a lower floor than PCCC at high SNR. Performance of these codes are shown in Fig. 5 through Fig. 9 and explained in Section 4.

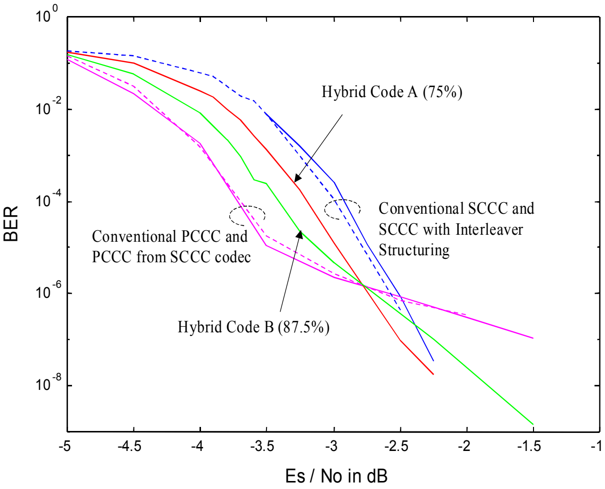


Figure 5: BER performance comparison of $r = \frac{1}{3}$ PCCC, SCCC and hybrid codes with frame size = 512 bits vs. E_s/N_0 in dB.

4 Simulation Results

Fig. 5 through Fig. 9 show simulation results of bit error rate performance of PCCC, SCC and hybrid codes against signal to noise rati (E_s/N_0) in dB for a number of different fram sizes: 512, 1022, 2048, 4096 and 8192 bits. Si curves are simulated in each case: convention PCCC, PCCC generated from a SCCC code conventional SCCC, SCCC with the interleave structuring and two hybrid codes.

In each case, the code polynomials used ar [35,23] (in octal), constraint length K = 5, linea log-MAP SISO decoders are used as constituen decoders, overall code rate is $r = \frac{1}{3}$ and the code bits are transmitted over an AWGN channel. The waterfall region of the curves is simulated for all frame sizes apart from 512 while the curves for frame size 512 are simulated down to the floor.

As expected, the SCCC code is worse at low

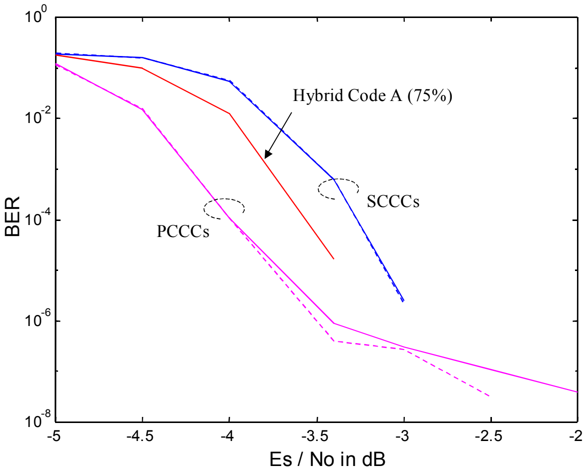


Figure 6: BER performance comparison of $r = \frac{1}{3}$ PCCC, SCCC and hybrid codes with frame size = 1022 bits vs. E_s/N_0 in dB.

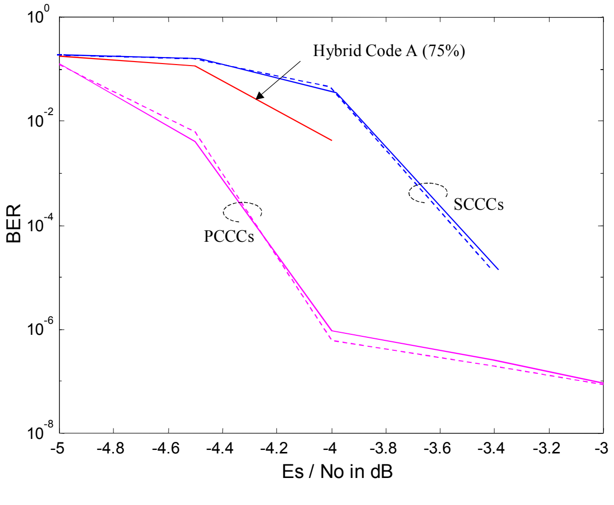


Figure 7: BER performance comparison of $r = \frac{1}{3}$ PCCC, SCCC and hybrid codes with frame size = 2048 bits vs. E_s/N_0 in dB.

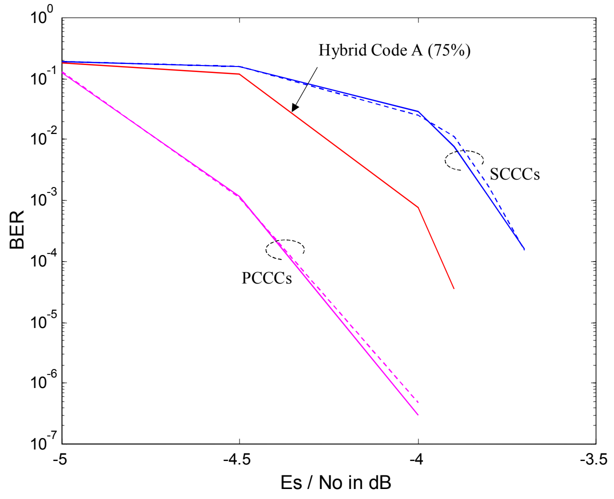


Figure 8: BER performance comparison of $r = \frac{1}{3}$ PCCC, SCCC and hybrid codes with frame size = 4096 bits vs. E_s/N_0 in dB.

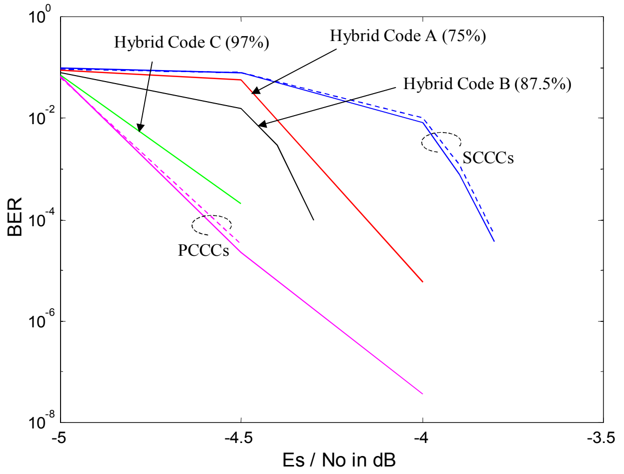


Figure 9: BER performance comparison of $r = \frac{1}{3}$ PCCC, SCCC and hybrid codes with frame size = 8192 bits vs. E_s/N_0 in dB.

SNR, but shows the lowest error rate floor, while the PCCC code is best at low SNR but has the highest floor. The double-parity in SCCC increases the minimum distance of the code and alleviates the occurrence of the high BER floor seen in PCCC. Two new hybrid code designs (A and B) are shown which serve as intermediate cases. These hybrid codes are constructed by puncturing the specified number of double-parity bits (either hybrid code A with 75% or hybrid code B with 87.5%) and then using single-parity bits from the Par_o/Sys_i field for the remaining parity bits (so that the overall code rate is maintained at $\frac{1}{3}$). The choice of the Par_o/Sys_i field for puncturing the remaining parity bits is due to the fact that puncturing the systematic bits of the outer encoder (also the input information bits) processed by the inner encoder gives rise to detrimental performance of the hybrid code.

It can be assumed that, since alternate parity bits of the inner encoder are punctured to generate the rate $\frac{1}{3}$ SCCC curves, on an average, 50% of the punctured parity bits are double-parity bits. A number of such hybrid codes can be generated with the number of punctured double-parity bits ranging between 50% (SCCC case) and 100% (PCCC case). The simulation results confirm that the 75% and 87.5% cases indeed "bridge the gap" between the SCCC and PCCC case, with the code containing fewer double parity bits (87.5% punctured) behaving closer to the PCCC and the code containing more double parity bits (75% punctured) behaving close to the SCCC.

We have used a puncturing period of 8 to puncture the double-parity bits while generating the hybrid codes. If a large puncturing period is used, then a varying percentage of double-parity bits can be punctured giving rise to an increased number of puncturing patterns. A large puncturing period can thus be used to generate multiple hybrid codes. Although the difference in performance between the individual hybrid codes will be marginal for small frame sizes, it is worthwhile to consider additional performance curves such as the hybrid code C with 97% double-parity bits punctured when large frame sizes are used.

5 Design and Analysis of Hybrid Codes

Parent codes comprised of good constituent codes can be designed using the two analysis and design approaches presented in [5] and [6] in order to yield better error rate performance. Furthermore, using these two approaches, the design of efficient puncturing schemes can also be achieved. The two design approaches we consider in this paper are: (1) Density evolution analysis [5] and (2) Mutual information and convergence theory [6]. However, our focus in this paper will be on the first approach.

Iterative decoding of turbo-like codes (PCCC, SCCC and hybrid codes) can be evaluated by tracking the density of extrinsic information messages passed between the constituent decoders during several iterations. There are two methods to track this density [5]: (1) Actual density evolution and (2) Gaussian approximation to the density evolution. We shall refer back to the parent code system model in Fig. 1 during the following description.

Analysis using actual density evolution of the constituent SISO decoders comprises of plotting histograms of the extrinsic information at the output of the previous decoder during each iteration ($\lambda(u;O)$ passed by the inner decoder and $\lambda(c;O)$ from the outer decoder). Values are then retrieved from the histograms to form the input extrinsic information to the next decoder ($\lambda(u;I)$ at the inner decoder and $\lambda(c;I)$ at the outer decoder). The input and output SNRs of extrinsic information for each decoder are computed from the λ-histograms as SNR $= E\{\lambda)\}/2$. These SNRs are then plotted - SNRout of decoder 1 is plotted against the SNRin of decoder 1 while SNRin of decoder 2 is plotted against SNRout of decoder 2.

It is seen that the decoder will converge to the correct codeword only if the two curves do not intersect. The improvement in SNR of extrinsic information and the associated BER improvement follows a staircase path reflecting at right angles. The space between the curves can be considered a tunnel (an iterative decoding tunnel). When the curves are very close to each other, the improvement in BER is rather slow and it takes many iterations to bore through the tunnel. Once the decoder passes through this tunnel, the curves begin to diverge indicating that the decoder will converge to the right codeword. However, if the curves intersect at any point, there will be no further improvement in SNR and the decoder will not converge.

The λ-histograms exhibit Gaussian-shaped probability densities when plotted. This is observed to be consistent in all turbo and turbo-like concatenations [5]. Thus, probability density function of extrinsic information (λ) can be approximated by a Gaussian density function with mean $\mu = E(\lambda)$ and variance $\sigma^2 = \mathrm{Var}(\lambda)$. If the λ-pdfs are assumed to be both Gaussian and symmetric, then the Gaussian density function depends only on its mean since $\sigma^2 = 2\mu$ and $\mathrm{SNR} = \mu/2$. Another benefit of the Gaussian approximation model is that while the actual density evolution model requires iterative processing of input and output extrinsic information, the Gaussian approximated model can be applied independently to individual constituent decoders for large interleaver sizes.

Another approach to understand the convergence behaviour of iterative decoders is to measure the entropy and mutual information between transmitted systematic bits and extrinsic information or a priori information. The mutual information transfer characteristics at the input and output of the corresponding decoders are plotted in an extrinsic information transfer (EXIT) chart. This analysis tool is especially useful in understanding the performance of turbo-like codes at low E_b/N_0.

Performance of hybrid codes can be analysed by using either the density evolution or the mutual information approach. As a result, good constituent codes and efficient puncturing schemes can be designed, and this, indeed will be the focus of our future work.

6 Conclusion

In this paper, a novel design of a parent code based on the SCCC model was presented that generates a family of hybrid codes bridging the gap between the PCCC and the SCCC. The design and implementation of the parent SCCC code was discussed. Further, error rate performance simulation curves of this code were presented, which exhibit very promising results. The hybrid codes can average the performance of the PCCC and the SCCC, and thus, perform well in all ranges (low as well as high) of signal to noise ratio. Finally, an insight into the techniques used to conduct design and analysis of such hybrid code designs was provided.

The proposed code is extremely flexible. System designers can benefit from this study since they will now have a broad range of coding options to choose from. Furthermore, the hybrid codes can be made to adapt to specific user, application and channel demands in next generation systems.

References

[1] Y. Wu and M. Valenti, "An ARQ technique using related parallel and serial concatenated convolutional codes," in *Proc., IEEE Int. Conf. on Commun.*, vol. 3, (New Orleans, LA), pp. 1390–1394, June 2000.

[2] C. Berrou, A. Glavieux, and P. Thitimasjshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes(1)," in *Proc., IEEE Int. Conf. on Commun.*, (Geneva, Switzerland), pp. 1064–1070, May 1993.

[3] S. Benedetto, G. Montorsi, D. Divsalar, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design, and iterative decoding," *IEEE Trans. Info. Theory*, vol. 44, pp. 909–926, May 1998.

[4] D. Divsalar and F. Pollara, "Hybrid concatenated codes and iterative decoding," *JPL TDA Progress Report*, vol. 42-130, pp. 1–23, April-June 1997.

[5] D. Divsalar, S. Dolinar, and F. Pollara, "Iterative turbo decoder analysis based on density evolution," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 891–907, May 2001.

[6] S. ten Brink, "Convergence of iterative decoding," *Electronics Letters*, vol. 35, pp. 806–808, May 24th 1999.

[7] S. Dolinar and D. Divsalar, "Weight distributions for turbo codes using random and non-random permutations," *JPL TDA Progress Report*, vol. 42-122, pp. 56–65, Aug. 15 1995.

[8] M. Valenti and J. Sun, "The UMTS turbo code and an efficient decoder implementation suitable for software defined radios," *Int. J. Wireless Info. Networks*, vol. 8, pp. 203–216, Oct. 2001.