Analysis of the Convergence Process by EXIT Charts for Parallel Implementations of Turbo Decoders

O. Sanchez, Christophe Jego, Michel Jezequel

Analysis of the Convergence Process by EXIT Charts for Parallel Implementations of Turbo Decoders

Oscar Sánchez*, Christophe Jégo† *Member IEEE* and Michel Jézéquel* *Member IEEE*

***Abstract*—Iterative process is a general principle in decoding powerful FEC codes such as turbo codes. However, the mutual information exchange during the iterative process is not easy to analyze and to describe. A useful technique to help the designer is the EXtrinsic Information Transfer (EXIT) chart. Unfortunately, this method cannot be directly applied to the decoding convergence analysis if parallel processing has to be exploited for the design of turbo decoders. In this letter, an extension of the EXIT charts method is proposed in order to take into account the constraints introduced by parallel implementations. The corresponding analysis associated with Monte-Carlo simulations gives additional understanding of the convergence process for the design of parallel architectures dedicated to turbo decoding.**

I. Introduction

TURBO coding [1] is a well-known channel-coding technique widely used in digital communications because of its error correcting capabilities close to the Shannon limit. It is especially attractive for wireless communications where several standards, such as UMTS, CDMA2000 or 3GPP-LTE, have adopted it. However, the design of high throughput Turbo Decoders (TDs) is a major challenge in today's systems. Indeed, the next generations of mobile communication systems will require data rates of 1 Gb/s and beyond. To respond to this challenge, designers have to exploit the maximum feasible amount of parallelism in TDs. The decoding of turbo codes is carried out through an iterative process where two Soft-Input Soft-Output (SISO) decoders, operating in natural and interleaved domain, exchange extrinsic information. SISO decoding algorithm is usually known as the Maximum A Posteriori (MAP), Forward-Backward or Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm. MAP-based algorithms such as Log-MAP and Max-Log-MAP [2] are the most common decoding algorithms for a hardware implementation. Unfortunately, these algorithms implement recursive operations making them difficult to parallelize.

Various parallelism techniques have been proposed in order to design high throughput TDs. In [3], a multi-level classification of parallel turbo decoding techniques with three hierarchical levels has been proposed: TD level parallelism, SISO decoder level parallelism and metric computation level parallelism. The first parallelism level corresponds to the duplication of the turbo decoder architecture. Since this level is not efficient in terms of hardware resources and frame decoding latency, it is not considered in this work. SISO decoder level parallelism consists in the use of several SISO decoders that work simultaneously. Two well-known techniques have been proposed. The first one is referred to as sub-block parallelism, and the second one as shuffled parallelism. In sub-block parallelism each received frame is divided into several sub-blocks. Then, each sub-block can be decoded in parallel. The basic idea of shuffled decoding is to decode both constituent convolutional codes simultaneously, exchanging extrinsic values as soon as possible [4]. The metric computation level parallelism exploits on one hand the parallelism of the trellis structure and, on the other hand, the parallelism of metric computation [5].

In all previous works, the BER performance degradation introduced by parallel processing is not explicitly considered and analyzed. Generally, to overcome the BER performance lost additional iterations are executed, with a corresponding negative impact on the TD throughput. A behavior analysis has to be performed to estimate the convergence of these techniques that increase the parallelism level. Moreover, new schedules have to be explored based on Monte-Carlo simulations associated to EXIT charts for designing decoding architectures suitable with high throughput. In this letter, we study the relation between different parallelism techniques and the additional TD iterations executed to prevent BER performance lost. An extension of the current EXIT chart technique is particularly proposed to enable the convergence analysis of the parallel decoding process.

This letter is organized as follows. Section II presents parallel techniques commonly used to increase the TD throughput. Section III introduce an extension of the EXIT chart method to analyze parallel TDs by adapting the usual method. In section IV parallel TD schemes are analyzed considering their transfer characteristics. A modified schedule for the computations inside the SISO decoders is also proposed.

II. Parallel turbo decoding approaches

We recall that metric computation and SISO decoder are the two parallelism levels considered due to their efficiency.

A. Metric computation parallelism level

Some parallelism can be achieved depending on the way the computations are executed by each SISO decoder. Note that we refer to schedule as the organization of the MAP-based algorithm computations. Fig. 1 depicts four different schedules

* Institut Telecom / Telecom Bretagne, CNRS Lab-STICC UMR 3192, Brest. Université Européenne de Bretagne, France. oscar.sanchez-gonzalez@telecom-bretagne.eu, michel.jezequel@telecom-bretagne.eu.

† IPB / Enseirb-Matmeca, CNRS IMS, UMR 5218, Bordeaux. Université de Bordeaux, France. christophe.jego@ims-bordeaux.fr.

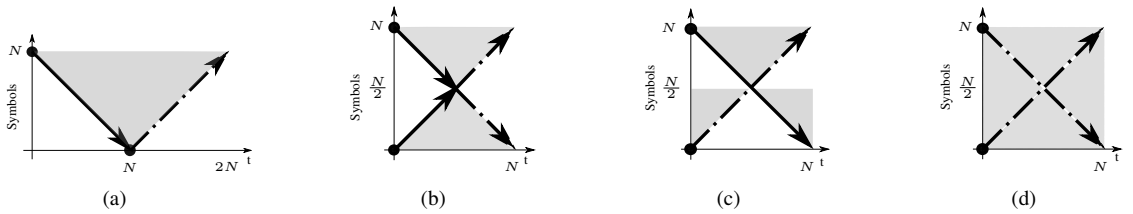


Fig. 1. Schedules for the MAP based algorithms: classical schedules ((a) Backward-Forward, (b) Butterfly) and schedules for shuffled turbo decoding ((c) Butterfly-Forward, (d) Butterfly-Replica).

for the decoding process over a sub-block of N information symbols. The horizontal axis represents the time and the vertical axis the current information symbol considered by the SISO decoder. One time unit is necessary for one transition in the trellis diagram. Continuous lines symbolize α or β computation and dashed lines state metrics with *a-posteriori* values ($L_{a,k}$) computation. Since in order to compute $L_{a,k}$, both forward state metrics α and backward state metrics β are necessary, some values have to be memorized. The shaded areas in Fig. 1 represents the time interval during which a memory has to store α or β values. Finally, black dots $\bullet$ correspond to initial values for the forward or backward state metrics computation by using the message passing method.

From the expression of the Forward-Backward algorithm, several schedules can be devised. A common schedule that produces soft output values in natural order is called Backward-Forward (B-F). This schedule is considered as reference in the remainder of this letter (Fig. 1(a)). Initially, β state metrics are computed recursively starting from the end until the beginning of a sub-block. β values are memorized. When all β values of the sub-block of size N are computed, α state metrics computation can start. In parallel, thanks to the β values previously computed, soft output values $L_{a,k}$ are calculated in natural order. In the Backward-Forward schedule, the decoding process duration is equal to twice the length N of a sub-block. This duration is halved by using the Butterfly schedule (Fig. 1(b)), at the cost of doubling the hardware resources to compute α, β and $L_{a,k}$. Schedules Butterfly-Forward and Butterfly-Replica in Fig. 1(c) and 1(d) respectively, have the same decoding duration as the Butterfly schedule. However, they generate *a-posteriori* values continuously. To this end, some state metrics values have to be stored at the end of each half iteration. Both schedules were defined to improve the convergence of shuffled TDs.

Let the 3-tuple $\Phi^{\xi}_{Q,S} = \{Q, S, \xi\}$ denotes the parallelism level of a TD architecture with $Q \geq 1$ the number of sub-blocks in the natural and interleaved domain, S the shuffled (*Sh*) or not shuffled (*NoSh*) parallelism, and ξ the decoding schedule - Backward-Forward (*B-F*), Butterfly (*B*), Butterfly-Forward (*B-FW*) or Butterfly-Replica (*B-R*) -. For convenience, we refer to this notation in the rest of the paper.

B. SISO decoder parallelism level

In order to exploit decoder parallelism, sub-blocks processing can be applied. The sub-blocks are processed in parallel in each domain and serially between the two domains, or in a full parallel manner. Since the computation of the state metrics α and β in the MAP-based algorithm are obtained by a recursive process, the sub-blocks are not independent. It means that the sub-block processing imposes a major constraint: the sub-block initialization of state metrics α and β. A survey of sub-block initialization can be found in [3]. The initialization could be achieved by applying three different methods: initialization by acquisition, initialization by message passing and initialization by combining the two previous methods. We have chosen the second method that requires minimal additional hardware resources and has a minor impact on BER performance. Intuitively, in the shuffled decoding approach each SISO component has faster access to update *a-priori* values. It would enable a faster convergence, *i.e.* fewer iterations are necessary to achieve the same BER performance with respect to non-shuffled (serial) decoding approach. Indeed, extrinsic values are generated for both domains during each half iteration when shuffled parallelism is applied, while in the non-shuffled approach during a half iteration the extrinsic information of only one domain is generated.

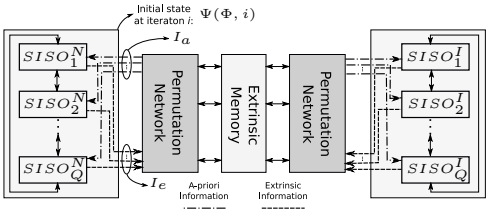


Fig. 2. Architecture based on shuffled and sub-block parallelisms.

Moreover, sub-block and shuffled parallelism techniques can be combined. Thus, several SISO decoders work simultaneously on different sub-blocks in the natural and interleaved domain. This approach is shown in Fig. 2, where $SISO^N_p$ and $SISO^I_p$ for $0 < p \leq Q$ are the decoders in the natural and interleaved domains respectively. An extrinsic memory is necessary to transfer extrinsic values between the decoders of the two domains. Two permutation networks are also necessary in order to provide interleaver and de-interleaver functions.

III. EXIT CHART METHOD EXTENSION

The EXtrinsic Information Transfer (EXIT) chart [6] is a useful kind of diagram proposed to analyze the convergence process of iterative decoding systems. Let us consider a TD architecture $\Phi^{\xi}_{1,NoSh}$ with no parallelism at the SISO decoder level. The transfer function that defines the relation between the mutual *a-priori* information at the input (I_a) and the mutual information at the output (I_e) of the sole SISO decoder in the architecture is given by (1) where E_b/N_0 is the Signal to Noise Ratio (SNR).

$$I_e = T(I_a, E_b/N_0) \quad (1)$$

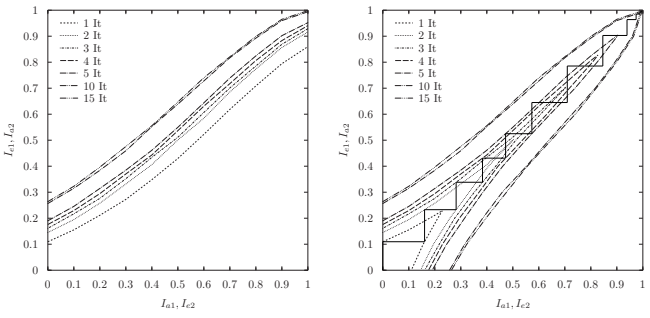
For a shuffled TD $\Phi^{\xi_m}_{1,Sh}$ with a schedule $\xi_m \in \{$*B-F*, *B*$\}$ the transfer function in (1) holds as well. In [7] a method to generate EXIT charts of shuffled TDs was introduced. Three Gaussian random noise generators are used: one for the channel and two others to generate the *a-priori* information. However, if a sub-block parallelism is applied in shuffled or not shuffled schemes, (1) is not valid anymore. Similarly, this equation cannot be applied to schedules *B-FW* and *B-R* with a shuffled parallelism. For these configurations, some values have to be kept during each half iteration in order to execute the next half iteration. It means that the transfer function has to take into account the dependency on additional parameters. Moreover, the EXIT chart method assumes that the extrinsic values have a Gaussian probability density function that has to satisfy the consistency condition. In this study, mean values of the EXIT band charts are considered to respect this constraint, as explained in [8].

Let $\Psi(\Phi, i)$ denote the state of a TD with a parallelism level Φ, at the beginning of the half iteration i. For a sub-block parallelism, $\Psi(\Phi, i)$ represents the initial state metric values α and β in the limits of the sub-blocks (dots $\bullet$ in Fig. 1). If schedules *B-FW* or *B-R* are considered, $\Psi(\Phi, i)$ also symbolizes α and β values inside the sub-blocks at the beginning of the decoding process of each half iteration (height of the shaded area at time $t = 0$ in Fig. 1(c) and 1(d)). Thus, the expression of the transfer function becomes:

$$I_e = T(I_a, E_b/N_0, \Psi(\Phi, i)) \quad (2)$$

For each value of E_b/N_0, we obtain a set of transfer curves that depends on the half iteration i. Fig. 3(a) shows the extrinsic information transfer characteristics for several iterations of the LTE TD with parallelism $\Phi^B_{128,NoSh}$ at $E_b/N_0 = 1dB$. Please note that the frame size is set to $L = 6144$ with a rate-$1/2$ code for all our experimentations. During the first iteration, the mutual information at the output is never equal to one, not even when the mutual information at the input is maximal. Iteration after iteration, the transfer characteristics improve so that the TD converges. When few iterations are performed the metrics in the limits of the sub-blocks after each iteration ($\Psi(\Phi^B_{128,NoSh}, i)$) are not good enough. Hence, even with a perfect knowledge of the information to decode ($I_a = 1$) at the input of the SISOs, the decision taken is wrong ($I_e < 1$). As more iterations are executed, better values for $\Psi(\Phi^B_{128,NoSh}, i)$ are obtained letting the TD to converge.

The decoding trajectory for the TD $\Phi^B_{128,NoSh}$ is given in Fig. 3(b). For convenience the TD characteristics are only shown for some iterations. Each transfer curve is plotted up to its first intersection. The decoding trajectory is thus plotted by "jumping" between the different transfer curves as the iterations are carried out. Thus, the EXIT chart or decoding trajectory of any parallel TD Φ can be plotted. In this way, convergence of any parallelism scheme can be analyzed.



(a) (b)

Fig. 3. a) EXIT chart for a SISO decoder. b) decoding trajectory. (Parallelism $\Phi^B_{128,NoSh}$, $E_b/N_0 = 1dB$, $L = 6144$ bits).

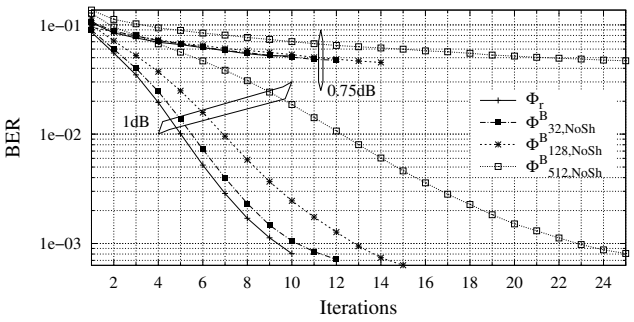


Fig. 4. Convergence of TDs with sub-block parallelism ($L = 6144$ bits).

IV. Turbo Decoding Convergence Analysis

Let us consider the architecture $\Phi_r = \Phi^B_{1,NoSh}$. The BER performance of this architecture after $I(\Phi_r) = 8$ iterations is taken as reference for an unconstrained turbo decoding This performance is smaller than 0.1dB compared to that achieved by an ideal turbo-decoding algorithm and is compliant with the LTE standard. Thus, we desire that any parallel TD Φ have similar BER performance after $I(\Phi)$ iterations (no BER performance lost introduced by the TD parallelism). Convergence of the TDs Φ_r, $\Phi^B_{32,NoSh}$, $\Phi^B_{128,NoSh}$ and $\Phi^B_{512,NoSh}$, at $SNR = [0.75dB, 1dB]$, are shown in Fig. 4. For $SNR = 1dB$, the number of iterations required for no BER performance degradation are $I(\Phi^B_{32,NoSh}) = 9$, $I(\Phi^B_{128,NoSh}) = 11$, $I(\Phi^B_{512,NoSh}) = 20$, respectively. Since the duration of an iteration is reduced when more sub-blocks are used, it is possible to increase the TD throughput. Convergence of shuffled TDs $\Phi^{\xi_n}_{1,Sh}$ with schedule $\xi_n \in \{B$, *B-FW, B-R*$\}$ is shown in Fig. 5. *B-R* schedule reduces significantly the number of iterations (almost to the half with respect to Φ_r). *B* and *B-FW* schedules have similar behaviors. However, the latter needs about one additional half iteration. For low SNR values both schedules behave almost identically.

EXIT charts of the different parallel TDs for whom convergence has been shown are presented in Fig. 6. The EXIT chart at $SNR = 0.75dB$ of the architecture Φ_r is depicted in Fig. 6(a). For this SNR value, the diagram starts just opening. It corresponds to the beginning of the "waterfall" region. The decoding trajectory of the TD $\Phi^B_{512,NoSh}$ for some iterations is plotted as well. The need of additional iterations can be noted. Fig. 6(b) shows the decoding trajectories of TDs with sub-block parallelism for $SNR = 1dB$. The decoding trajectory

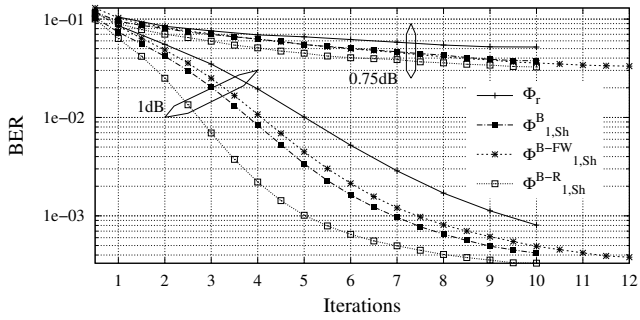


Fig. 5. Convergence of shuffled TDs ($L = 6144$ bits).

of $\Phi^{B}_{32,NoSh}$ is very close to the one that would follow Φ_r. For thirteen iterations, the BER performance of $\Phi^{B}_{512,NoSh}$ is worse than that of $\Phi^{B}_{32,NoSh}$ with six iterations, which is also worse than that of $\Phi^{B}_{128,NoSh}$ with eight iterations. Coherent results between Fig. 4 and Fig. 6(b) confirm the correctness of the decoding trajectories that have been obtained in Eq. (2).

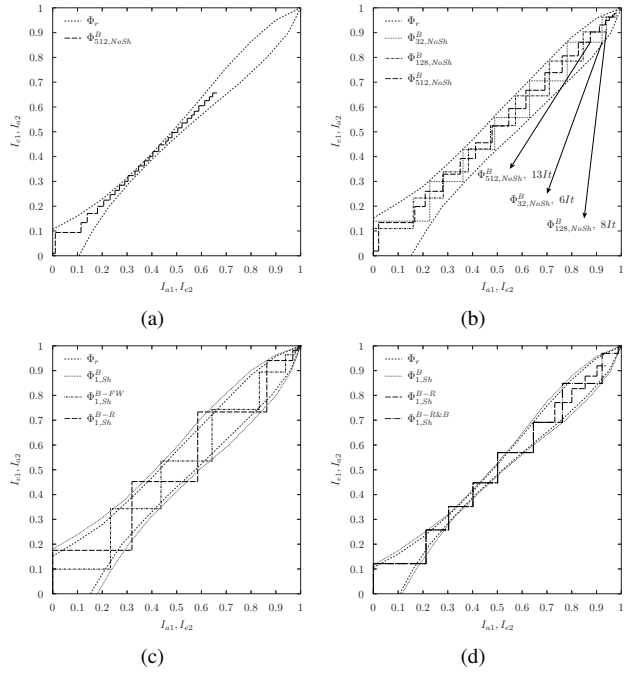


Fig. 6. a) Decoding trajectory for $\Phi^{B}_{512,NoSh}$ at $0.75dB$. b) Decoding trajectories for 32, 128 and 512 sub-blocks no shuffled TDs. c) Decoding trajectories for shuffled TDs at $1dB$. d) Decoding trajectories for schedules *B-R*, *B* and a combination of both at $0.75dB$.

Shuffled TDs EXIT charts for $SNR = 1dB$ are presented in Fig. 6(c). EXIT chart of $\Phi^{B}_{1,Sh}$ is wider than the EXIT chart of Φ_r. For the *B-R* schedule the decoding trajectory exceeds that of the Butterfly schedule in shuffled mode ($\Phi^{B}_{1,Sh}$). Even if $\Phi^{B}_{1,Sh}$ and $\Phi^{B\text{-}R}_{1,Sh}$ behave similarly after the first half iteration, the decoding trajectory of the *B-R* schedule moves forward faster. $\Phi^{B\text{-}FW}_{1,Sh}$ presents the worst characteristics to reduce the number of iterations. This occurs mainly due to the poor behavior after the first half iteration. Since during this half iteration β values of the first half of the sub-block are unknown (Fig. 1(c)), the *a-posteriori* values generated during the decoding of this part of the sub-block are not appropriate. After the first half iteration, all values α and β of the sub-block are properly calculated, and the decoding trajectory of the schedule *B-FW* tries to approach that of the EXIT chart of $\Phi^{B}_{1,Sh}$. However, the decoding trajectory cannot reach that of the Butterfly schedule and keeps at about one half iteration lower. This behavior does not appear for $\Phi^{B\text{-}R}_{1,Sh}$. Indeed for the schedule *B-R*, the *a-posteriori* values are computed with appropriate α and β values computed at each half iteration.

Since for shuffled parallelism the decoding is wider, shuffled TDs reach the region of waterfall at lower SNR values with respect to non-shuffled TDs. Fig. 6(d) illustrates how the decoding trajectory with schedule *B-R* can easily converge where the EXIT chart of Φ_r becomes tight. Even more, for $\Phi^{B}_{1,Sh}$ the EXIT chart is slightly wider than the EXIT chart of Φ_r. It can be seen however that the decoding trajectory of $\Phi^{B\text{-}R}_{1,Sh}$ narrows for high mutual input information.

When schedule *B-R* is used, at the beginning of each half iteration α and β values inside the sub-block (height of the shaded area in Fig. 1(d) at $t = 0$) correspond to the initial state of the TD. If those values are not good enough, they might prompt decoding errors even though the mutual information at the input of the SISO decoder is high. This behavior is similar to the one observed in Fig. 3(a), where not so good α and β values in the sub-block limits lead to errors in the early iterations for high input mutual information. No similar result is observed for $\Phi^{B}_{1,Sh}$ since no initial values are kept from iteration to iteration. By taking into account this result, a new schedule can be proposed. It consists in employing *B-R* schedule during the early iterations, and then switching to Butterfly schedule. The decoding trajectory obtained for this new schedule (*B-R*&*B*) is presented in Fig. 6(d) where after 5 iterations the schedule is switched from *B-R* to *B*. Indeed, during the first 5 iterations the decoding trajectories of $\Phi^{B\text{-}R}_{1,Sh}$ and $\Phi^{B\text{-}R\&B}_{1,Sh}$ are similar. Thus, fast convergence can be achieved, and since the hardware resources are the same for both schedules, no additional hardware complexity is required.

V. Conclusion

Parallel turbo decoding techniques have been analyzed by observing the mutual information transfer characteristics of SISO decoders. Since the usual approach to plot EXIT charts cannot be directly applied, a key modification to this kind of diagrams has been proposed. In this way, EXIT charts become a useful method to understand the convergence of all parallel TDs. Moreover, a new schedule has been proposed for the shuffled TDs. Advantages of shuffled parallelism in order to reduce the number of iterations and to reach the waterfall region earlier have been observed. An integrated turbo decoder dedicated to LTE turbo codes has now been designed based on the current analysis for a 1 Gb/s experimental setup.

References

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo codes," *IEEE International Conference on Communications*, pp. 1064–1070, 1993.

[2] P. Robertson, E. Villebrun, and Hoeher, "A comparison of optimal and suboptimal map decoding algorithms operating in the log domain," *IEEE Int. Conf. Communications*, pp. 1009–1013, 1995.

[3] O. Muller, A. Baghdadi, and M. Jezequel, "Exploring parallel processing levels for convolutional turbo decoding," *Information and Communication Technologies. ICTTA 06. 2nd*, pp. 2353 – 2358, 2006.

[4] J. Zhang and F. M.P.C, "Shuffled iterative decoding," *IEEE Trans. on Communications*, vol. 53, pp. 209–213, 2005.
[5] G. Masera, G. Piccinini, M. Roch, and M. Zamboni, "Vlsi architectures for turbo codes," *Very Large Scale Integration (VLSI) Systems, IEEE Trans. on*, vol. 7, no. 3, pp. 369 –379, sept. 1999.
[6] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. on Communications*, vol. 49, pp. 1727–1737, 2001.
[7] J. Zhang, Y. Wang, and M. Fossorier, "Iterative decoding with replicas," *IEEE Trans. on Information Theory*, vol. 53, pp. 1644–1663, 2007.
[8] J. Lee and R. Blahut, "Lower bound on ber of finite-length turbo codes based on exit characteristics," *Communications Letters, IEEE*, vol. 8, no. 4, pp. 238 – 240, april 2004.