Reducing the Complexity of LDPC Decoding Algorithms: An Optimization-Oriented Approach

Sarajlic, Muris; Liu, Liang; Edfors, Ove

Published in:
2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)

DOI:
10.1109/PIMRC.2014.7136286

2015

Reducing the Complexity of LDPC Decoding Algorithms: An Optimization-Oriented Approach

Muris Sarajlić, Liang Liu, and Ove Edfors
Department of Electrical and Information Technology, Lund University, Sweden
Email: (muris.sarajlic, liang.liu, ove.edfors)@eit.lth.se

***Abstract*—This paper presents a structured optimization framework for reducing the computational complexity of LDPC decoders. Subject to specified performance constraints and adaptive to environment conditions, the proposed framework leverages the adjustable performance-complexity tradeoffs of the decoder to deliver satisfying performance with minimum computational complexity. More specifically, two constraint scenarios are studied: the "good-enough" performance and "as-good-as-possible performance". Moreover, we also investigate the effects of different degrees of freedom in performance-complexity tradeoff adjustments. The effectiveness of the proposed method has been verified by simulating a set of LDPC codes used in IEEE 802.11 and IEEE 802.16 standards. Computational complexity reductions of up to 35% have been observed.**

I. Introduction

Low Density Parity Check (LDPC) codes have established themselves as a channel coding technique of choice for a number of contemporary communication standards, like WiMAX (IEEE 802.16e), IEEE 802.11n and other [1], [2]. The strength of LDPC codes primarily lies in their ability to approach the information transmission capacity of the communication channel, which proves to be of great importance in communication systems with tight bandwidth efficiency and performance constraints.

The high error-correcting capability comes with the price of high computational complexity, which is a critical issue in real-life implementations of LDPC decoders, especially in mobile handsets where the high performance has to be handled with limited battery power and silicon area. This has driven constant research efforts aiming at reducing the complexity of the decoding algorithms. Basic complexity reduction techniques for the decoding algorithm focus on substituting the mathematical functions in the original iterative belief-propagation/sum-product (BP/SP) algorithm [3] with their approximations [4]. The properties of the original decoding algorithm (and its modifications) itself offer an array of possibilities for complexity reduction. It has been observed [5] that the iterative decoding process can be stopped after a certain predefined number of iterations I_{max}, as further iterations do not bring any performance improvement. In addition to this, a wide spectrum of techniques (jointly referred to as *early termination* techniques) can be employed that will, at each iteration, perform a check which will indicate whether further iterations are necessary and use this information to stop the decoding process at $I < I_{max}$ iterations [5], [6], [7], [8]. Finally, in an early termination variant referred to as *forced convergence* [9], [10] the iterative updating of individual codeword bits is stopped if they are determined to be converged. If the convergence criterion is properly set, the forced convergence can yield complexity reduction with minimum performance loss.

Although the schemes listed are elaborate and are shown to yield good results, there is still room for further complexity reduction. To achieve this target, we propose to fully exploit the potential provided by the adjustable decoding parameters with a structured optimization framework. The basic idea is that the parameter adjustment is adaptive to channel conditions, but in such a way so that the decoder always delivers satisfactory performance at minimum possible computational cost.

In this paper, the described optimization-oriented approach is applied in two study cases:

1) LDPC decoding algorithm that employs the forced convergence technique in a system with relaxed performance constraints (more specifically, $FER = 10^{-2}$) and uses optimum design parameters at each SNR point. The result is a *channel-adaptive* decoding algorithm that uses the performance margin to produce complexity reduction;
2) In order to meet tighter performance requirements, the described decoder is paired up with an auxiliary decoder that corrects the errors produced by forced convergence. Optimization is utilized here to find the point where the joint complexity of the two decoders is minimum and it is shown that the obtained complexity reduction is approximately the same as in case 1.

II. Background

A. General considerations

LDPC codes [3] are linear block codes with codeword length N described fully by a sparse parity check matrix $\boldsymbol{H}$ with dimensions $M \times N$. Their structure can be conveniently represented in the form of a bipartite graph [11] where codeword bits are represented by bit (variable) nodes and parity checks by parity (check) nodes, with their interconnections mapped directly from the parity check matrix. The process of decoding of LDPC codes can be viewed as iterative message passing between adjacent variable and check nodes which perform processing of the incoming messages. In this work,

978-1-4799-4912-0/14/$31.00 © 2014 IEEE

a low-complexity approximation of the BP/SP algorithm, the offset min-sum (OMS) algorithm [4] is chosen for the analysis.

The order in which messages are passed from variable nodes to check nodes and vice versa (message passing scheduling) directly affects the performance. In the simplest scheduling scheme (referred to as *flooding* scheduling) all v-nodes simultaneously pass the messages to their adjacent c-nodes, and the same goes for c-nodes. In the *layered* scheduling scheme [6], [12], c-nodes and their adjacent v-nodes are grouped in *layers*, and the exchange of messages between v-nodes and c-nodes is done for each layer separately, in a sequential fashion. This strategy is shown to have multiple advantages over the flooding scheduling, the main ones being reducing the number of iterations needed for the algorithm to converge, and reduced memory access [12].

In order to describe the layered OMS algorithm, let us denote aposteriori v-node LLRs by Q_v, apriori v-node LLRs (obtained from the channel) by P_v, extrinsic messages passed from v to c by $Q_{temp,vc}$ and extrinsic messages passed from c to v by R_{cv}. Additionally, let $N(c)$ denote the set of v-nodes adjacent to a c-node c, $X(\cdot)$ a hard bit decision operator (converting Q_v to 0 or 1) and ω an offset value. Finally, $\boldsymbol{Q} = [Q_1 \quad Q_2 \ldots Q_N]$ is the codeword LLR vector.

B. Forced convergence: theoretic background

If the behaviour of individual Q_v is observed over iterations i through I_{max}, it can be seen how, for SNRs after the "turbo cliff", most of the Q_v evolve towards $+\infty$ or $-\infty$, corresponding to the decoder being more and more "confident" about them being 0 or 1. It can then be reasonable to assume that the updating of Q_v can be stopped after their magnitude crosses some predefined threshold θ. Depending on the value of θ, this will introduce a certain performance degradation, but will also introduce savings in complexity because all operations leading to the update of Q_v can be stopped. The described principle is referred to as forced convergence.

The OMS algorithm with layered scheduling that applies forced convergence is presented in Fig. 1. The offset ω is introduced with the goal of improving the performance, and the hard syndrome check performed after each iteration is an early termination technique aiming at reducing the average number of iterations.

```
for all v-nodes v and c-nodes c do                ▷ The initialization section
    R_cv ← 0
    Q_v ← P_v
end for
Inact = ∅
for iterations i to I_max do                      ▷ The iteration section
    for all layers k do
        for all c in layer k do
            for all v ∈ N(c) do
                Q_temp,vc ← Q_v − R_cv
            end for
            Q_min1 ← min_{v∈N(c)} |Q_temp,vc|
            Q_min2 ← min_{v∈N(c), v≠v_min1} |Q_temp,vc|
            S = ∏_{v∈N(c)} sign(Q_temp,vc)
            for all v ∈ N(c)  AND  v ∉ Inact do
                if v = v_min1 then
                    R_cv ← sign(Q_vc) · S · max{Q_min2 − ω, 0}
                else
                    R_cv ← sign(Q_vc) · S · max{Q_min2 − ω, 0}
                end if
                Q_v ← Q_temp,vc + R_cv
                if |Q_v| > θ then
                    |Q_v| ← θ
                    v :∈ Inact
                end if
            end for
        end for
    end for
    if H · X(Q^T) = 0 then
        stop iterations
    end if
end for
```

Fig. 1: OMS with layered scheduling, parity checking and forced convergence

III. Optimization-Oriented Approach to Decoding Algorithm Complexity Reduction

A. General problem formulation

We start by a simple formulation of the communication receiver system design problem (which can be easily applied to any system design problem in engineering). The values of variable system design parameters are given in the vector $\boldsymbol{\delta}$, values of fixed design parameters by vector $\boldsymbol{\sigma}$ and the effects of the environment (reliability indicators) by vector $\boldsymbol{\rho}$. Values in $\boldsymbol{\sigma}$ and $\boldsymbol{\rho}$ are outside of the influence of the designer. Receiver design problem can then be formulated as

$$\underset{\boldsymbol{\delta}}{\text{minimize}} \quad \Gamma(\boldsymbol{\delta}, \boldsymbol{\sigma}, \boldsymbol{\rho}) \tag{1a}$$

$$\text{subject to} \quad \Pi(\boldsymbol{\delta}, \boldsymbol{\sigma}, \boldsymbol{\rho}) \geq \Pi_{con} \tag{1b}$$

where $\Gamma(\cdot)$ and $\Pi(\cdot)$ are functions modeling system complexity and performance respectively, and Π_{con} is the target performance. Receiver design then boils down to finding the solution $\boldsymbol{\delta_{opt}}$ for the optimization problem (1). In this work, two forms of Π_{con} are analyzed:

(a) $\Pi_{con} = \Pi_{max}(\boldsymbol{\sigma}, \boldsymbol{\rho}) - \epsilon$,

(b) $\Pi_{con} = \Pi_{relax}, \quad \Pi_{relax} << \Pi_{max}(\boldsymbol{\sigma}, \boldsymbol{\rho})$.

The constraint case (a) allows only a very small deviation ϵ from the maximum performance Π_{max} achievable with given $\boldsymbol{\sigma}$ and $\boldsymbol{\rho}$ (from now on, this constraint will be referred to as "tight"). Constraint case (b), on the other hand, allows for a (usually generous) performance margin (and will henceforth be referred to as "relaxed").

To put this structured problem formulation to the test, we analyze two case studies of reduced-complexity decoder design, both based on forced convergence.

B. Reduced-complexity decoding, case study 1: Forced convergence with relaxed performance constraint (RPC)

For the first case study, we take the design of a decoding algorithm that uses forced convergence and "relaxed" performance constraint and model it in the form of the optimization problem (1). It can be quickly seen that $\boldsymbol{\delta} = \theta$, all the fixed design parameters (decoding algorithm OMS, modulation, code rate, etc.) are incorporated in σ and we take $\rho = SNR$. We adopt the average computational complexity (average number of computations needed for decoding a codeword) $\bar{C}$ as $\Gamma(\cdot)$, and average frame error rate $\overline{FER}$ as $\Pi_{con}(\cdot)$. If we denote the computational complexity of case study 1 with $\bar{C}_t(\theta, SNR)$ and the corresponding FER as $\overline{FER}_1(\theta, SNR)$, the design problem for a particular system configuration $\boldsymbol{\sigma}$ can be formulated as

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \bar{C}_t(\theta, SNR) \tag{2a}$$

$$\text{subject to} \quad \overline{FER}_1(\theta, SNR) \leq FER_{relax} \tag{2b}$$

The resulting $\theta_{opt} = f(SNR)$. In order to solve problem (2), $\bar{C}_t(\theta, SNR)$ needs to be obtained by means of analytical complexity analysis of the algorithm in Fig. 1. We start by emphasizing that in the forthcoming analysis "operations" denotes additions and comparisons; the impact of XORs on the total complexity is neglected. The additions and comparisons are further considered to have the same complexity, so in the end the number of additions is taken as the complexity measure. We would like to point out that a complete and precise complexity model would have to take into account all the details of the actual implementation (such as bit wordlength) and especially memory access activity. The model presented here is approximate and it mostly serves the general purpose of demonstrating how optimization-oriented approach can be used in algorithm design.

The number of v-nodes that perform operations in one iteration (i.e. that are "visited" in the decoding process) is random due to applying forced convergence. We therefore define the average number of nodes "visited" by the algorithm in iteration i as $\bar{A}(i, \theta, SNR)$; we additionally denote the maximum possible number of visited nodes in an iteration (property of matrix $\boldsymbol{H}$) as A_{max}. By analyzing the flow of the algorithm, it can be observed that the operations in one iteration can be split into two sets: the ones whose number is independent of θ (in lines 9-14) and the ones affected by θ (corresponding to lines 15-26). The total number of operations in one iteration can then be expressed as

$$\alpha + \#op \cdot \bar{A}(i, \theta, SNR), \tag{3}$$

where $\alpha = A_{max} + |c| \cdot \Psi$. $|c|$ here denotes the number of c-nodes, and $\Psi = \sum_m p_m(m + \lceil \log_2 m \rceil - 2)$, where p_m is the fraction of c-nodes c having order equal to m. $m + \lceil \log_2 m \rceil - 2$ is the complexity of finding two minimum elements in an array of length m [13]. $\#op$ is the number of operations (additions) in the algorithm section affected by θ and is in this case equal to 4.

Since the algorithm employs syndrome checking after each iteration, the number of iterations performed is random. If we denote the probability of iteration i happening as $p_t(i, \theta, SNR)$, the average computational complexity for all iterations is calculated as

$$\begin{aligned} &\bar{C}_t(\theta, SNR) \\ &= \sum_{i=1}^{I_{max}} p_t(i, \theta, SNR)[\alpha + 4\bar{A}(i, \theta, SNR)] \\ &= \alpha \sum_{i=1}^{I_{max}} p_t(i, \theta, SNR) + 4 \sum_{i=1}^{I_{max}} p_t(i, \theta, SNR)\bar{A}(i, \theta, SNR) \\ &= \alpha \bar{N}_{t,it}(\theta, SNR) + 4\bar{N}_{t,nodes}(\theta, SNR) \end{aligned} \tag{4}$$

Function $\bar{N}_{t,it}(\theta, SNR)$ represents the average number of iterations, and $\bar{N}_{t,nodes}(\theta, SNR)$ is the average number of visited nodes for all iterations. These functions, together with $\overline{FER}_1(\theta, SNR)$, need to be determined by means of Monte-Carlo simulations.

In this work, complexity reduction is always measured with respect to the OMS algorithm that employs hard syndrome checking but does not employ forced convergence (henceforth referred to as "reference case"). For this case, the operations corresponding to the comparison with the threshold are saved, but on the other hand there is no reduction of the number of visited v-nodes and A_{max} nodes are always visited at each iteration. The total number of operations performed at each iteration is equal to $4 \cdot A_{max} + |c| \cdot \Psi$. Since hard syndrome checking is performed, the number of iterations is still random but now only depends on the SNR. If the probability of iteration i is denoted as $p_n(i, SNR)$, the average computational complexity for all iterations can be found as

$$\begin{aligned} \bar{C}_n(SNR) &= [4 \cdot A_{max} + |c| \cdot \Psi] \sum_{i=1}^{I_{max}} p_n(i, SNR) \\ &= \gamma \ \bar{N}_{n,it}(SNR). \end{aligned} \tag{5}$$

Percent savings for the optimum threshold can then be expressed as

$$S_{opt}(SNR) = \frac{\bar{C}_n(SNR) - \bar{C}_{t,opt}(SNR)}{\bar{C}_n(SNR)} \cdot 100\%. \tag{6}$$

C. Reduced-complexity decoding, case study 2: Forced convergence with tight performance constraint (TPC)

We now consider a system that uses forced convergence as the design parameter but with a tight performance constraint. In order to meet stricter performance requirements, an auxiliary decoder can be employed that will redo the decoding of blocks that are erroneous due to forced convergence. What is initially unclear in this setup is how much can the complexity of the primary decoder be reduced and how does this reflect to the overall complexity of the decoder pair. However, it can be observed that, as the complexity of the primary decoder is reduced, the number of errors that need to be corrected is increased and thus the complexity of the auxiliary decoder increases. The optimization approach can then be used to find the point at which the joint complexity of the two decoders is minimized.

The complexity of the primary decoder using forced convergence is $\bar{C}_t(\theta, SNR)$, the same as in (4), the complexity of the

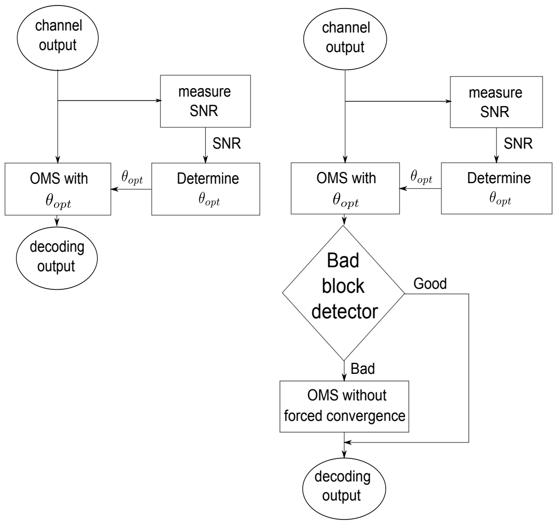

Fig. 2: Operation principle of decoder algorithms in case studies 1 (left) and 2 (right)

auxiliary decoder is denoted as $\bar{C}_a(\theta, SNR)$ and additionally there is a "bad block" detector with complexity $\bar{C}_{det}(\theta, SNR)$. The design of the minimum complexity decoder can now be formulated as

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \bar{C}_t(\theta, SNR) + \bar{C}_a(\theta, SNR) + \bar{C}_{det}(\theta, SNR) \tag{7a}$$

$$\text{subject to} \quad \overline{FER}_2(\theta, SNR) \leq FER_{min,\boldsymbol{\sigma}}(SNR) + \epsilon \tag{7b}$$

The diagrams representing the operational principles of decoding algorithms analyzed in case studies 1 and 2 are shown in Fig. 2.

The optimization problem (7) can be reformulated following the observation that a large majority of erroneous blocks generated by forced convergence decoding doesn't converge to a valid codeword and still have active v-nodes at I_{max}. Using the fact that at the SNRs of interest, $\overline{FER}_1(\theta, SNR) >> FER_{min,\sigma}(SNR)$, it can be concluded that a large majority of erroneous blocks are erroneous because of forced convergence and the process of their decoding will last until I_{max}; in other words, with high probability we can claim that *block with active v-nodes at* $I_{max} \Leftrightarrow$ *block in error due to forced convergence* $\Leftrightarrow$ *block in error*. The detection of an erroneous block is trivial in this case: it's enough to check the value of the iteration counter, and turn the auxilliary decoder on when $i = I_{max}$. With this decoder design, the complexity of the primary decoder stays the same as in (7), and the complexity of the auxiliary decoder is multiplied with the probability of it being engaged, equal to $\overline{FER}_1(\theta, SNR)$. Given this model, the performance constraint is satisfied and becomes a part of the cost function, transforming the optimization problem into an unconstrained one. With the complexity of the "bad block" detector neglected, the reformulated design problem is now

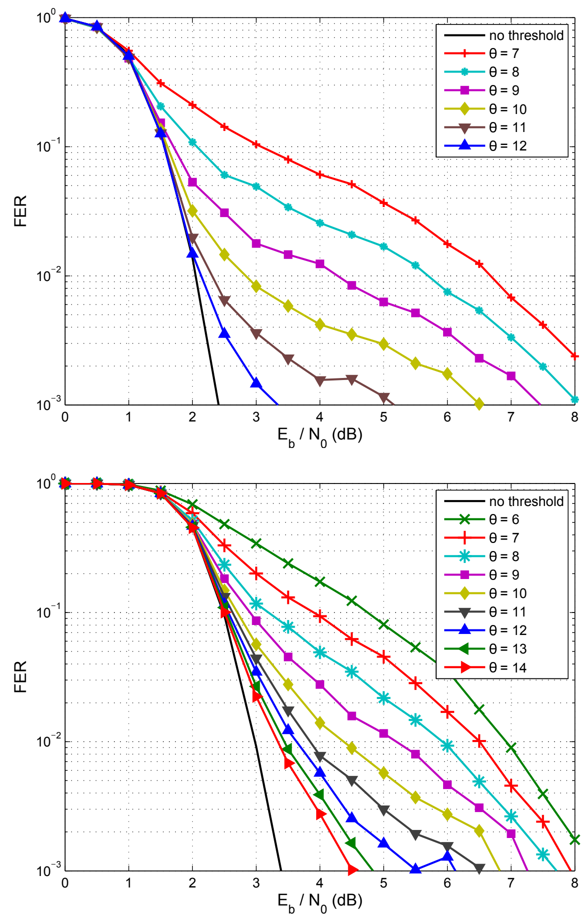

Fig. 3: FER for different thresholds, upper: code 1, lower: code 2

given as

$$\underset{\boldsymbol{\theta}}{\text{minimize}}\ \bar{C}_t(\theta, SNR) + \overline{FER}_1(\theta, SNR) \cdot \bar{C}_a(\theta, SNR), \tag{8}$$

where the auxiliary decoder performs the basic OMS algorithm with no forced convergence and $\bar{C}_a(\theta, SNR) = \gamma \sum_{i=1}^{I_{max}} p_a(i, \theta, SNR)$. The average number of iterations for the auxiliary decoder is in general not equal to $p_n(i, SNR)$ and is dependent on θ because the fraction of blocks that will not converge to a valid codeword in any case (with or without forced convergence) is higher for the auxiliary decoder, yielding $\sum_{i=1}^{I_{max}} p_a(i, \theta, SNR) > \sum_{i=1}^{I_{max}} p_n(i, SNR)$ for a given (θ, SNR) point.

IV. Simulations and Results

The concepts presented were tried out with a set of LDPC codes, given in Table I. Codes are taken from IEEE 802.16e and IEEE 802.11n standards [1], [2]. The modulation used is QPSK and the channel is AWGN. Due to large simulation times, the simulations were performed on a coarse (SNR, θ) grid with steps $0.5\ dB$ for SNR and 1 for θ; in order to obtain more detailed results, the obtained data was then interpolated over a finer grid using cubic interpolation, and

TABLE I: LDPC codes used in the simulations

Code 1	IEEE 802.11n, R = 1/2, N = 648
Code 2	IEEE 802.16e, R = 2/3 (B), N = 672
Code 3	IEEE 802.16e, R = 1/2, N = 672
Code 4	IEEE 802.16e, R = 2/3 (B), N = 2304

the optimizations were performed on this interpolated data. For the sake of brevity, only the results for codes 1 and 2 are given here; however, the trends that were observed apply to the other two codes as well.

We start by presenting in Fig. 3 the FER curves for different values of θ and the reference case. FER curves show the effect θ has on the performance: the evolution of the magnitude of Q_v is stopped and the magnitude fixed at a certain value, possibly also fixing the erroneous sign of Q_v and causing a reduction of flow of information to other v-nodes in subsequent iterations. This results in a performance degradation and increase of the error floor; the performance degradation increases with progressively smaller values of θ. The FER results can be practically used in solving (2): if $\overline{FER}_1(\theta, SNR) = FER_{relax}$ is solved for θ, a new function $\theta_{con}(SNR)$ is obtained, and the constraint (2b) can then be rewritten as $\theta \geq \theta_{con}(SNR)$. In the performed analysis, $FER_{relax} = 10^{-2}$, which is a common performance constraint in contemporary wireless applications. This constraint also serves for determining a range of SNRs (SNR_{min}, SNR_{max}) in which the optimization is performed, with SNR_{min} being approximately equal to the point where FER for the reference case is equal to 10^{-2}.

The level plots of $\bar{C}_t(\theta, SNR)$ and $\bar{C}_t(\theta, SNR) + \overline{FER}_1(\theta, SNR) \cdot \bar{C}_a(\theta, SNR)$ are shown in Fig. 4, overlaid with $\theta_{con}(SNR)$ and $\theta_{opt}(SNR)$ for case study 1 and $\theta_{opt}(SNR)$ for case study 2. The data shown is for code 1; the observed patterns apply to other codes as well. In line with the preceding discussion, for case study 1, $\bar{C}_t(\theta, SNR)$ is minimized in the half-plane above $\theta_{con}(SNR)$.

It can be observed that both cost functions have large values for large and small values of θ, and the minima lie in between. With large θ, $\bar{N}_{t,nodes}(\theta, SNR) \rightarrow A_{max} \cdot \bar{N}_{n,it}(SNR)$, or simply, the node activity tends to the maximum one, corresponding to the reference case. On the other hand, $\bar{N}_{t,it}(\theta, SNR)$ increases with decreasing θ, causing the increase of the cost function. It can be further noted how the behaviour of $\theta_{opt}(SNR)$ in case 1 is "shaped" by the constraint, whereas it remains approximately constant over the operating SNR range for case study 2 (this behaviour is also observed for other tested codes).

The analysis continues with comparing the complexity of decoding algorithms shown in Fig. 2 using $\theta_{opt} = f(SNR)$ with the variants of the same algorithms that would use a θ_{opt} that is constant over the SNR region of interest. For study case 1, this constant θ_{opt} value can again be obtained by solving (2), now with an added constraint $SNR_{min} \leq SNR \leq SNR_{max}$. In study case 2, a constant θ that best approximates θ_{opt} can be chosen after solving (8). These suboptimum threshold values are denoted as $\theta_{subopt,RPC}$ and $\theta_{subopt,TPC}$ for case studies 1 and 2, respectively. The values of $\theta_{subopt,RPC}$ and $\theta_{subopt,TPC}$ for code 1 (corresponding to optimum values shown in Figure 4) are 12.2 and 10, respectively.

Fig. 5 shows the decoding complexities, expressed as the total number of additions normalized by I_{max} and number of information bits, and savings in complexity compared to the reference case, as defined by (6), for codes 1 and 2. The first thing that can be observed is that the complexity for all cases reduces with increased SNR which is a consequence of the syndrome checking mechanism which can be viewed as an *implicit* SNR adaptation scheme. The complexity can be further reduced by employing forced convergence with a suboptimum θ that is constant over the operating SNR range. The maximum complexity reduction, however, is achieved by constantly adapting θ to the SNR, that is, by employing *explicit* adaptation of θ to channel conditions in addition to the implicit one. Furthermore, it can be observed that the decoder algorithm in case study 2 almost reaches the complexity of the decoder in case study 1, in spite of a more demanding performance constraint; this is of course made possible by introducing another degree of freedom, namely the auxiliary decoder. From all the analyzed cases, it follows that applying $\theta_{subopt,TPC}$ is the same as applying $\theta_{opt,TPC}$, that is, channel adaptation in this case doesn't introduce additional complexity reduction.

V. Conclusion

In this work, a structured, optimization-oriented approach to the design of LDPC decoding algorithms has been demonstrated. It has been shown how the analytical model for the algorithm complexity can serve as the cost function, and various design constraints as constraints in an optimization problem. By comparing various solutions of the design problem, corresponding to different design constraints, system designers can gain a deeper insight in the tradeoffs involved in system design. Applying the optimization-oriented approach on the design of an LDPC decoder algorithm employing forced convergence, it was demonstrated that, given a relaxed performance constraint, the largest complexity reduction is achieved by constant adaptation of the threshold to the SNR, and that a non-adaptive threshold provides a suboptimal complexity reduction. It was also demonstrated how introducing another design parameter - cleanup of block errors, a tight performance constraint can be met although forced convergence is still used, and the optimization approach here provides the value of the thresholds that optimally balances the complexity of the primary and auxiliary decoders.

Acknowledgment

The work presented is a part of the Digitally Assisted Radio Evolution (DARE) project, and the authors would like to thank the Swedish Foundation for Strategic Research (SSF) for providing the funds for the project.

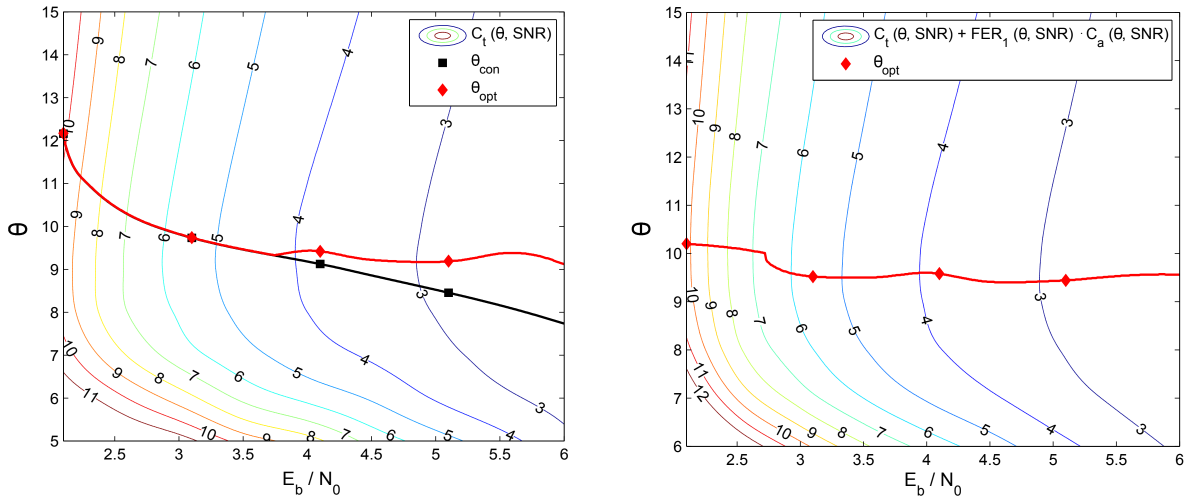

Fig. 4: Left: threshold constraint and optimum threshold for case study 1, code 1. Right: optimum threshold for case study 2, code 1. $FER_{relax} = 10^{-2}$

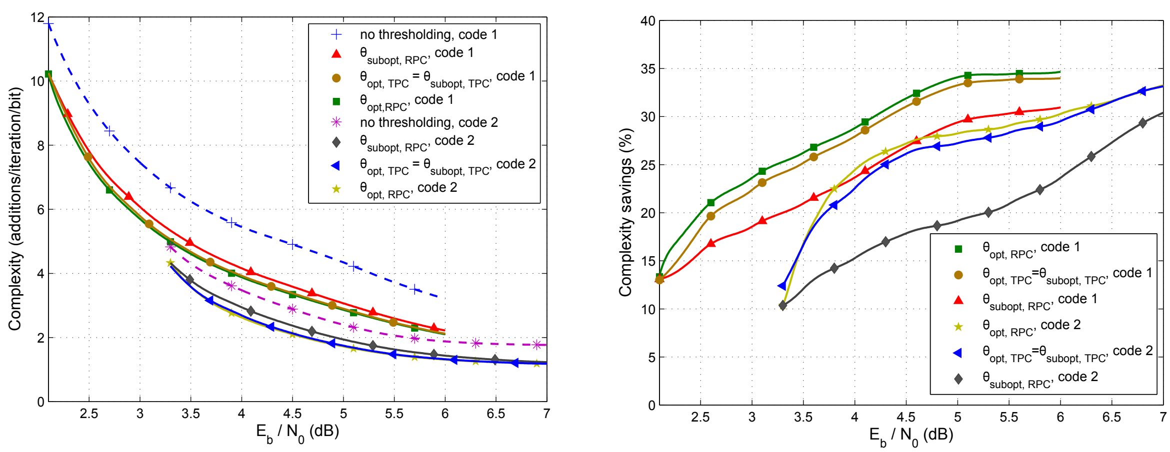

Fig. 5: Left: decoding complexity, right: decoding complexity savings, for both case studies and both codes

References

[1] IEEE 802.16-2009, IEEE Standard for Local and metropolitan area networks, part 16: Air Interface for Broadband Wireless Access Systems [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=5062428

[2] IEEE 802.11n-2012, IEEE Standard for Information Technology-Telecommunications and information exchange between systems, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications [Online]. Available: http://standards.ieee.org/getieee802/download/802.11-2012.pdf

[3] R. G. Gallager, *Low-Density Parity-Check Codes.* Cambridge, MA: MIT Press, 1963.

[4] J. Chen, A. Dholakia, E. Eleftheriou, M.P.C. Fossorier, and X.Y. Hu, "Reduced-Complexity Decoding of LDPC Codes," *IEEE Tran. Comm.*, vol.53, no.8, pp.1288-1299, Aug. 2005.

[5] A. Darabiha, A.C. Carusone, and F.R. Kschischang, "Power Reduction Techniques for LDPC Decoders," *IEEE J. Solid-State Circuits*, vol.43, no.8, pp.1835-1845, Aug. 2008.

[6] D.E. Hocevar, "A reduced complexity decoder architecture via layered decoding of LDPC codes," *IEEE Workshop on Signal Process. Syst., 2004.*, pp.107-112, Oct. 2004.

[7] D. Shin, K. Heo, S. Oh, and J. Ha, "A Stopping Criterion for Low-Density Parity-Check Codes," *IEEE 65th Veh. Technology Conf., 2007.* pp.1529-1533, April 2007.

[8] F. Kienle and N. Wehn, "Low complexity stopping criterion for LDPC code decoders" *IEEE 61st Veh. Technology Conf., 2005.*, vol.1, pp.606-609 June 2005.

[9] E. Zimmermann, P. Pattisapu, P. K. Bora, and G. Fettweis, "Reduced Complexity LDPC Decoding Using Forced Convergence," *Proc. 7th Int. Symp. on Wireless Personal Multimedia Commun.*, Padova, Italy, Sep. 2004, vol. 3, pp. 243246, WA2-2.

[10] E. Zimmermann, P. Pattisapu, and G. Fettweis, "Bit-flipping post-processing for forced convergence decoding of LDPC codes," *Proc. 13th European Signal Process. Conf.*, Antalya, Turkey, Sep. 2005.

[11] R.M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol.27, no.5, pp.533-547, Sep. 1981.

[12] E. Sharon, S. Litsyn, J. Goldberger, "An efficient message-passing schedule for LDPC decoding," *Proc. 23rd IEEE Conv. of Elec. and Electron. Engineers in Israel, 2004.*, pp.223-226, Sept. 2004.

[13] P. V. Ramanan, L. Hyafil, "New algorithms for selection", *J. of Algorithms*, vol. 5, no. 4, pp. 557-578, Dec. 1984.