Improved Decoding of Binary and Non-Binary LDPC Codes by Probabilistic Shuffled Belief Propagation

Moritz Beermann, Laurent Schmalen, Peter Vary
Inst. of Comm. Systems and Data Processing
RWTH Aachen University, Germany
{beermann|schmalen|vary}@ind.rwth-aachen.de

***Abstract*—Low-density parity-check (LDPC) codes have proved to be very powerful channel coding schemes with a broad range of applications. However, as maximum-likelihood decoding is utterly complex, suboptimal decoders have to be employed. One of the most popular decoding algorithms of LDPC codes is belief propagation (BP) decoding. In this paper, we present a novel scheduling for belief propagation decoding of LDPC codes. The new approach combines probabilistic scheduling with the known shuffled and check-shuffled serial scheduling algorithms. The resulting probabilistic shuffled and probabilistic check-shuffled decoders show a superior performance in terms of residual bit error rate. The drawback is that the convergence speed is slightly decreased. However, the convergence is still faster than for the standard and probabilistic flooding algorithms. Furthermore, we have adapted the probabilistic and the proposed probabilistic serial BP schemes to the non-binary case. We show that the aforementioned effects on the binary decoder can similarly be observed when applying the different schedules to the decoding of LDPC codes over higher order Galois fields.**

***Index Terms*—LDPC Codes, Belief Propagation, Decoder Scheduling**

I. Introduction

Shortly after the breakthrough of the iterative decoding principle in Turbo codes [1], *Low-Density Parity-Check* (LDPC) codes, originally presented in [2], were rediscovered in [3]. LDPC codes are usually decoded using the *belief propagation* (BP) decoding algorithm [4], also called *sum-product* algorithm. BP decoding operates on the Tanner graph representation of the LDPC code. The Tanner graph is a bipartite graph consisting of variable nodes and check nodes. Even though BP decoding is not optimal in terms of maximum likelihood decoding, a performance close to the Shannon limit is observed with long LDPC codes.

The standard BP decoder uses a so-called *flooding* schedule: first the information for all the check nodes is evaluated and then "flooded" to the variable nodes. Then all the variable nodes are evaluated and the resulting information is flooded back to the check nodes. Several different scheduling approaches for improving the BP performance have been presented in the literature. A probabilistic schedule has been introduced in [5]. This approach takes into account the local girth (i.e., the length of the shortest cycle passing through a given node) of each node. To each variable node a probability depending on the local girth is assigned and at each iteration (except the first), the outgoing messages of each variable node are updated with this probability whereas outgoing check node messages are always updated. This approach improves the decoding performance of LDPC codes in terms of residual bit error rate (BER), however, the convergence properties slightly change for the worse.

In order to improve the convergence properties of BP decoding, manifold serial schedules have been presented in the literature, e.g. [6], [7], [8], [9]. Two examples are *shuffled* BP decoding [7] and the dual *check-shuffled* BP decoding [9]. These algorithms exploit the fact that new information obtained while updating single nodes can immediately be used in the updates of other nodes within the same iteration. This can lead to a faster decoding convergence and thus to a better BER performance in the case of a constrained number of iterations.

Other serial decoding schedules include metric-based approaches [10]. Here the problem of finding the best sequence of node updates is addressed by computing a metric for each message in the Tanner graph which causes a non-negligible computational overhead. Furthermore, the greediness of these approaches might lead to significantly higher error-floors after decoding, which is why they are not considered here.

In this work, we combine the girth-based probabilistic approach [5] with the shuffled [7] and check-shuffled [9] approaches and show that the performance in terms of BER can be improved compared to other serial scheduling strategies with (almost) no additional operations. The drawback is a slight loss in convergence speed, however, the convergence is still faster than for the standard BP decoding with simple flooding and the standard probabilistic parallel approach.

In [11] and [12] the effects of a serial updating scheme on the decoding of non-binary LDPC codes over GF(q) were examined. It was shown that similarly to the binary case the decoding process can be accelerated by a factor of up to two. We also present results for the adaptation of the probabilistic approach [5] and the proposed new scheduling to the decoding of non-binary LDPC codes over higher order Galois fields.

II. Belief Propagation Decoding

First, we will shortly review belief propagation (BP) decoding. The (N, K) LDPC code (length N, K information symbols) is characterized by a parity check matrix $\mathbf{H}$ of dimension

This work was supported by the research cluster UMIC funded by the German research council DFG.

$M \times N$ (with $M = N - K$). H_{mn} denotes the entry of $\mathbf{H}$ at row m and column n. The set $\mathcal{N}(m) = \{n : H_{mn} \neq 0\}$ denotes the symbols that participate in parity check equation m, $m = 1, \ldots, M$. Similarly, the set $\mathcal{M}(n) = \{m : H_{mn} \neq 0\}$ contains the check equations in which symbol n participates. Exclusion is denoted by the operator "\", e.g., $\mathcal{M}(n) \backslash m$ describes the set $\mathcal{M}(n)$ with check equation m excluded.

Let Z_n denote the received channel-related L-value for variable node n. Before starting the iterative processing, we set $v_{mn}^{(0)} = Z_n$ ($v_{mn}^{(i)}$ denotes the information at the edge connecting variable node n with check node m at iteration i). The first step of the BP decoder consists in determining the check-node related information

$$c_{mn}^{(i)} = 2 \tanh^{-1} \left(\prod_{n' \in \mathcal{N}(m) \backslash n} \tanh \left(\frac{v_{mn'}^{(i-1)}}{2} \right) \right) \tag{1}$$

with c_{mn} denoting the information at the edge connecting check node m with variable node n and i denoting the iteration counter. The second step updates the variable node information according to

$$v_{mn}^{(i)} = Z_n + \sum_{m' \in \mathcal{M}(n) \backslash m} c_{m'n}^{(i)} . \tag{2}$$

Additionally, for each variable node n, the hard decision $\hat{y}_n^{(i)} = \mathrm{sign}\{v_{mn}^{(i)} + c_{mn}^{(i)}\}$ (for an arbitrary m) is computed. Using those values the parity check equations are evaluated to determine whether the decoding can be stopped.

III. Novel Decoder Scheduling

We propose a novel scheduling strategy that takes into account the distribution of cycles in the Tanner graph on the one hand and on the other hand performs the updating of nodes in a serial fashion.

As pointed out in [5] it can be advantageous to consider the structure of the Tanner graph of a code in the decoding process. Since practically all Tanner graphs of LDPC codes contain cycles, the BP decoder optimality is violated. Let g_n denote the local girth of variable node n (i.e. the length of the shortest cycle passing through node n) and g_{min} and g_{max} the length of the shortest respectively the longest cycle in the Tanner graph. After $g_n/2$ iterations the information in a cycle of length g_n is propagated back to its origin and the assumption of statistical independence of the incoming extrinsic knowledge and the own knowledge does no longer hold. By only updating node n with probability $p_n^{[\mathrm{g}]} = g_n / g_{\text{max}}$ in each iteration the violation of the statistical independence can on average be postponed.

It has been shown that by serially updating messages in the decoding process the convergence speed of the BP decoder can be up to twice as fast as for the flooding strategy. This can be explained by the faster spreading of information when performing serial node updates [7].

A. Probabilistic Shuffled Scheduling

Our first scheduling strategy follows the concept of the *shuffled* schedule presented in [7] and adds a probabilistic component similar to the one presented in [5]. In a pre-processing step the local girth g_n of each variable node n is calculated and divided by the maximal local girth g_{max} to yield the updating probability $p_n^{[\mathrm{g}]} = g_n / g_{\text{max}}$. All variable-to-check messages $v_{mn}^{(0)}$ are initially set to the channel-related L-values Z_n and all check-to-variable messages $c_{mn}^{(0)}$ are set to zero. At the beginning of the ith iteration all check-to-variable messages $c_{mn}^{(i-1)}$ and all variable-to-check messages $v_{mn}^{(i-1)}$ are available as result of the previous iteration. In each iteration the decoder traverses all variable nodes in a serial fashion and randomly decides (using, e.g., a small lookup table with precomputed random numbers) whether to update the incoming and outgoing messages of each node or not with probability $p_n^{[\mathrm{g}]}$.

The probabilistic shuffled scheduling can be described by:

1) Initialize $n = 1$ and $i = 1$
2) With probability $1 - p_n^{[\mathrm{g}]}$ set $c_{mn}^{(i)} = c_{mn}^{(i-1)}$ and $v_{mn}^{(i)} = v_{mn}^{(i-1)}$ for all $m \in \mathcal{M}(n)$ and continue at step 5
3) Compute all incoming messages of variable node n (horizontal step), i.e. $\forall m \in \mathcal{M}(n)$ process:

$$c_{mn}^{(i)} = 2 \tanh^{-1} \left(\prod_{\substack{n' \in \mathcal{N}(m) \backslash n \\ n' < n}} \tanh \left(\frac{v_{mn'}^{(i)}}{2} \right) \cdot \prod_{\substack{n' \in \mathcal{N}(m) \backslash n \\ n' > n}} \tanh \left(\frac{v_{mn'}^{(i-1)}}{2} \right) \right)$$

4) Compute all outgoing messages of variable node n (vertical step), i.e. $\forall m \in \mathcal{M}(n)$ process $v_{mn}^{(i)}$ according to equation (2)
5) If $n < N$ increment n and continue at step 2
6) If stopping criterion is fulfilled abort decoding and output hard decision $\hat{y}_n^{(i)}$ for all n
7) Start next iteration by setting $i = i + 1$ and $n = 1$ and continue at step 2

Note that the iteration counter i does not correspond to the number of iterations in a flooding or a simple serial schedule, since – depending on the girth structure of the code – this schedule performs fewer message updates per iteration. To allow a fair comparison of the different schedules we count the number L_{up} of total message updates and specify the number of equivalent (flooding) iterations needed by a schedule as $I_{\text{sched}} = \left\lfloor \frac{L_{\text{up}}}{2E} + \frac{1}{2} \right\rfloor$, with E denoting the number of edges in the Tanner graph and $\lfloor x \rfloor$ denoting the largest integer smaller than or equal to x (fair rounding).

Note that the formal description of this schedule is similar to the *lazy scheduling* approach presented in [13]. However, in *lazy scheduling* an updating probability is chosen dynamically according to some reliability measure to avoid unnecessary message computations and reduce the overall complexity of the decoding process. Our approach, by contrast, uses fixed (girth dependent) updating probabilities and accepts a slight increase in the overall decoding complexity while it aims at achieving lower residual bit error rates.

B. Probabilistic Check-Shuffled Scheduling

Our second scheduling strategy can be described as a dual version of the probabilistic shuffled scheduling, iterating over the check nodes rather than the variable nodes, just as the scheduling in [9] is described as a dual version of the shuffled schedule in [7]. In accordance with Sec. III-A let g_n denote the local girth of variable node n and $p_n^{[g]} = g_n/g_{\max}$ the updating probability of variable node n. $\mathcal{P}^{(i)}$ denotes the set of variable nodes that participate in updates of iteration i. After the same initialization as in Sec. III-A at the beginning of each iteration the decoder decides for each variable node whether it is active or not (i.e. whether it sends and revceives updated messages) with probability $p_n^{[g]}$. Then all check nodes are traversed serially and incoming and outgoing messages are only updated for those edges that are connected to active variable nodes. This probabilistic check-shuffled scheduling can be described as follows:

1) Initialize $m = 1$ and $i = 1$
2) For $1 \leq n \leq N$: with probability $p_n^{[g]}$ add n to the set $\mathcal{P}^{(i)}$
3) $\forall n \in \mathcal{N}(m) \cap \mathcal{P}^{(i)}$: compute (vertical step):
$$v_{mn}^{(i)} = Z_n + \sum_{\substack{m' \in \mathcal{M}(n)\backslash m \\ m' < m}} c_{m'n}^{(i)} + \sum_{\substack{m' \in \mathcal{M}(n)\backslash m \\ m' > m}} c_{m'n}^{(i-1)}$$
4) $\forall n \in \mathcal{N}(m) \backslash \mathcal{P}^{(i)}$: set $v_{mn}^{(i)} = v_{mn}^{(i-1)}$
5) $\forall n \in \mathcal{N}(m) \cap \mathcal{P}^{(i)}$: compute (horizontal step):
$$c_{mn}^{(i)} = 2 \tanh^{-1} \left(\prod_{n' \in \mathcal{N}(m) \backslash n} \tanh \left(\frac{v_{mn'}^{(i)}}{2} \right) \right)$$
6) $\forall n \in \mathcal{N}(m) \backslash \mathcal{P}^{(i)}$: set $c_{mn}^{(i)} = c_{mn}^{(i-1)}$
7) If $m < M$ increment m and continue at step 3
8) If stopping criterion is fulfilled abort decoding and output hard decision $\hat{y}_n^{(i)}$ for all n
9) Start next iteration by setting $i = i + 1$ and $m = 1$ and continue at step 2

Again the total number of message updates L_{up} is counted and the number of equivalent (flooding) iterations I_{sched} is determined as described in Sec. III-A to allow a fair comparison to other schedules.

C. Adaptation to Decoding of Non-Binary LDPC Codes

The difference between the Tanner-graph representation of binary LDPC codes over GF(2) and non-binary LDPC codes over GF($q = 2^p$) is that in the non-binary case a weight according to the value of the corresponding check matrix element is assigned to each edge of the (weighted) Tanner-graph. Still, the definition of the local girth as the length of the shortest cycle passing through a node is the same in both cases since it is independent of the edge weights. Furthermore, the presence of (short) cycles causes the same degradation on the decoding performance due to the statistical dependence of messages after a certain number of decoding iterations. With this analogy it can be expected that the additional probabilistic component taking into account the girth structure of a code [5] has the same effect on the decoding of both binary and non-binary LDPC codes.

Results of [11] and [12] indicate that applying serial updating schemes to the decoding of non-binary LDPC codes has the same advantage of increasing the convergence speed as for the binary case.

We have investigated the effects of the probabilistic schedule and the proposed probabilistic shuffled schedule as introduced in Sec. III-A if adapted to the non-binary case. For this purpose we modified the log-FFT algorithm for the decoding of LDPC codes over GF(q) as presented in [14] and described in detail in [15] to allow for the employment of different schedules. Results for the combination of the log-FFT algorithm with flooding, probabilistic, shuffled and probabilistic shuffled schedules are presented in Sec. IV-B.

IV. Simulation Results

A. Binary LDPC Codes

In order to evaluate the performance of both proposed approaches for the binary case, we consider the WiMAX (576,288) rate-1/2 LDPC code. As a data source, we employ a Bernoulli source emitting equiprobable bits. A frame of $K = 288$ bits is encoded using the LDPC code. The resulting bits are mapped to BPSK symbols and then transmitted. On the channel, the (modulation) symbols (with symbol energy $E_{\text{s}} = 1$) are subject to additive white Gaussian noise (AWGN) with (known) power spectral density $\sigma_n^2 = N_0/2$.

Figure 1 depicts the BER as a function of E_{b}/N_0 ($E_{\text{b}} = 2E_{\text{s}}$). It can be seen that the proposed novel probabilistic shuffled and probabilistic check-shuffled algorithms outperform the other four schedules (flooding, probabilistic, shuffled, and check-shuffled).

Figure 2 shows the convergence at $E_{\text{b}}/N_0 = 2.5\,\text{dB}$. As expected, the convergence of the new schedules is slightly slower than for the shuffled approaches [7], [9] (however it still outperforms the flooding and probabilistic schedules). For

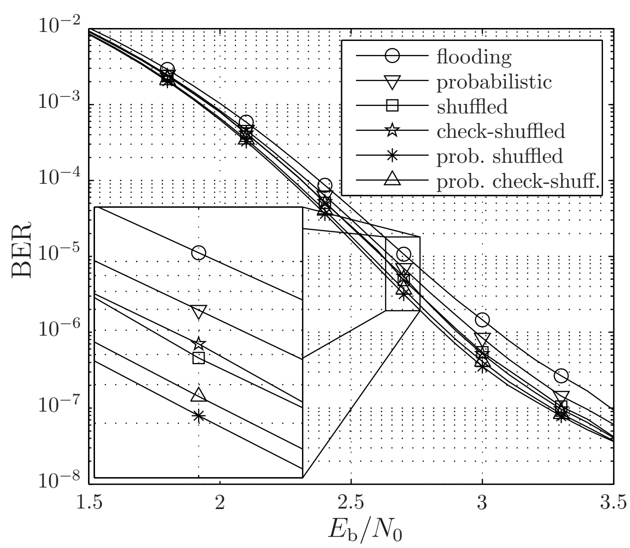

Fig. 1. BER performance of different BP-based decoding algorithms for the WiMAX (576,288) rate 1/2 LDPC code at 200 decoding iterations.

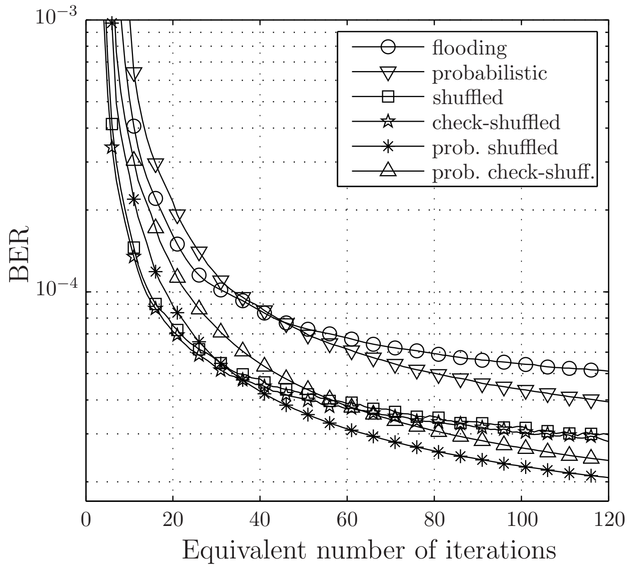

Fig. 2. Convergence (BER as a function of equivalent iterations i or I_{sched}) of different BP-based decoding algorithms at the fixed $E_{\text{b}}/N_0 = 2.5\,\text{dB}$ for the WiMAX (576,288) rate $1/2$ LDPC code.

more than 30 iterations, the probabilistic shuffled schedule outperforms the shuffled schedule and yields the lowest BER.

In Fig. 3 the average number of decoding iterations needed for convergence is plotted against the channel quality for the different schedules (solid lines). The maximum number of iterations is 40. The dashed lines show the ratio of the number of iterations needed by the flooding schedule and the number of iterations needed by the other schedules. It can be seen that the proposed schedules require slightly more (equivalent) iterations than the shuffled schedules but the saving compared to the flooding schedule is still as high as $28\,\%$.

Although the probabilistic check-shuffled schedule performs worse than the probabilistic shuffled schedule in this example, we have in some cases observed a better performance if using codes constructed by *progressive edge growth* (PEG) [16]. Furthermore, the advantages of the check-shuffled approach indicated in [9] hold. Finally note that strategies like group shuffling [7] (allowing parallelization) can be directly integrated into the proposed novel schedules.

B. Non-Binary LDPC Codes

For the evaluation of the proposed schedules for the non-binary case, two PEG Tanner-graph codes over GF(q) as presented in [16] are considered. The simulation setup is the same as described in Sec. IV-A. In accordance with Sec. III-C the modified log-FFT algorithm is used for the decoding of the non-binary LDPC codes. Results of flooding, probabilistic, shuffled and the novel probabilistic shuffled schedules are shown for two codes over GF(8) and GF(64) with rate $1/2$ and a binary block length of $K = 504$ information bits. The results for the check-shuffled and the proposed probabilistic check-shuffled schedules are not shown here since they are similar to those of the shuffled and probabilistic shuffled approaches.

Figure 4 shows the BER as a function of E_{b}/N_0 with a maximum number of 100 decoding iterations. Similarly to the binary case the proposed probabilistic shuffled schedule outperforms the three other schedules for both codes and for all shown channel qualities.

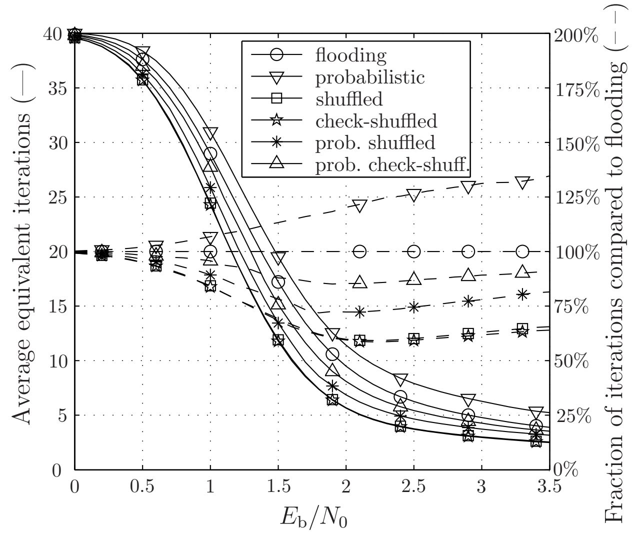

Fig. 3. Average number of equivalent iterations (i or I_{sched}) (solid lines) and fraction of iterations compared to the flooding schedule (dashed lines) needed by different BP-based decoding algorithms for the WiMAX (576,288) rate $1/2$ LDPC code. The maximum number of decoding iterations is set to 40.

In Fig. 5 the convergence at $E_{\text{b}}/N_0 = 1.9\,\text{dB}$ is depicted. Again a behavior similar to the binary case can be observed. The initial convergence speed is slightly higher for the non-probabilistic schedules (flooding and shuffled) but after a certain number of iterations lower bit error rates can be achieved by the probabilistic schedules. For more than 35 (resp. 40) iterations the proposed probabilistic schedule yields the lowest bit error rate for the code over GF(8) (resp. GF(64)).

Finally, Fig. 6 shows the average number of equivalent iterations needed by the modified log-FFT decoding algorithm

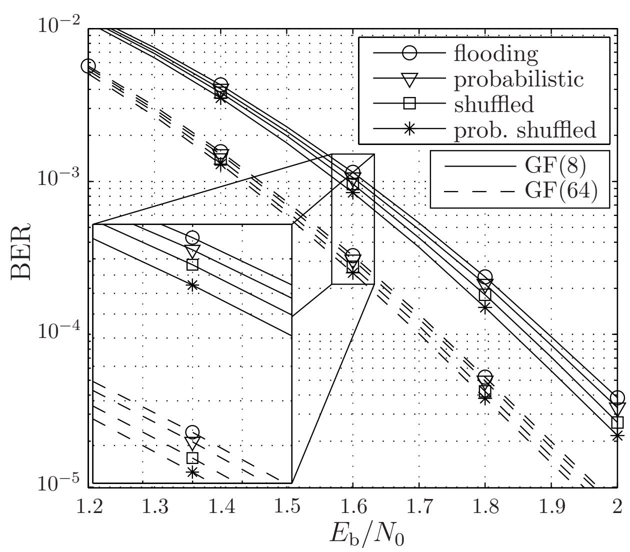

Fig. 4. BER performance of the modified log-FFT decoding algorithm using different schedules for the decoding of two non-binary rate $1/2$ LDPC codes over GF(8) and GF(64) with binary block length $K = 504$ using at most 100 decoding iterations.

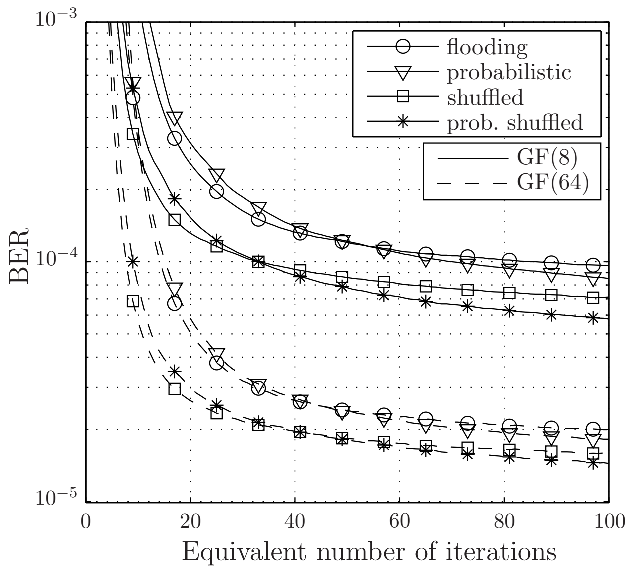

Fig. 5. Convergence (BER as a function of equivalent iterations i or I_{sched}) of the modified log-FFT decoding algorithm at the fixed $E_{\text{b}}/N_0 = 1.9$ dB using different schedules for the decoding of two non-binary rate $1/2$ LDPC codes over GF(8) and GF(64) with binary block length $K = 504$.

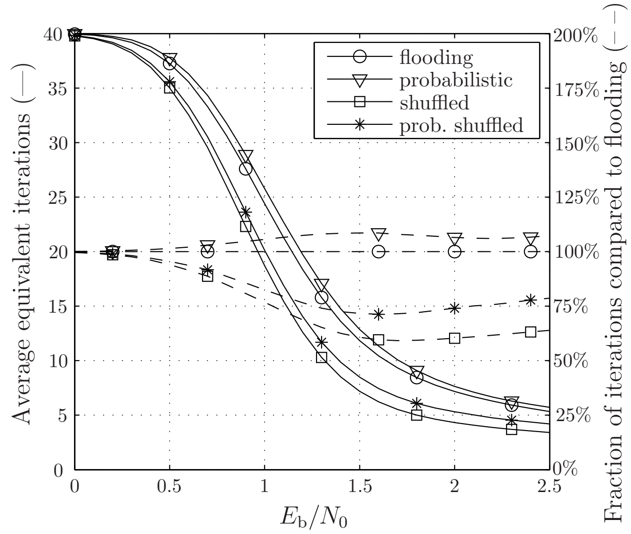

Fig. 6. Average number of equivalent iterations (i or I_{sched}) (solid lines) and fraction of iterations compared to the flooding schedule (dashed lines) needed by the modified log-FFT decoding algorithm using different schedules for the non-binary rate $1/2$ LDPC code over GF(8). The maximum number of decoding iterations is set to 40.

using different schedules for varying channel qualities E_{b}/N_0. For the sake of clarity the results of the code over GF(64) have been omitted in this figure. However, they show the same characteristics as those for the GF(8) code with the difference that the absolute number of iterations needed is slightly lower for the GF(64) code. The proposed schedule needs slightly more (equivalent) iterations than the shuffled schedule but still up to $28\,\%$ can be saved compared to the flooding schedule.

V. Conclusion

In this paper, we have shown how to improve the performance of belief propagation decoding of LDPC codes by using a probabilistic girth-based serial scheduling approach. Depending on the girth, a node is updated according to a certain probability: nodes with a large girth are updated more often than nodes with smaller girth. If this girth-based approach is combined with a shuffled serial decoding scheme, a superior performance in terms of BER is obtained. For the depicted WiMAX code, the BER is more than halved at $E_{\text{b}}/N_0 = 2.5$ dB compared to the standard flooding BP decoding with 120 iterations. Although the initial convergence speed is slightly lower in comparison with the serial algorithms, it is always faster than in the case of the pure probabilistic and the flooding algorithms. The proposed algorithm has (almost) no additional computational complexity compared to the (check-)shuffled schedules. The performance of the proposed schedules has been demonstrated for the LDPC code employed in the WiMAX system. Similar results were observed for a range of other codes. Furthermore, adaptations of the proposed novel schedules to the log-FFT decoding of non-binary LDPC codes over higher order Galois fields have been presented. It was shown by a range of simulation results that the effects of the different schedules on the decoding of non-binary LDPC codes are very similar to the binary case

References

[1] C. Berrou and A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo-Codes," *IEEE Trans. Commun.*, vol. 44, no. 10, pp. 1261–1271, Oct. 1996.

[2] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA, USA: M.I.T. Press, 1963.

[3] D. J. C. MacKay, "Good Error-Correcting Codes Based on Very Sparse Matrices," *IEEE Trans. Inf. Theory*, vol. 45, pp. 399–431, Mar. 1999.

[4] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA, USA: Morgan Kaufmann, 1988.

[5] Y. Mao and A. H. Banihashemi, "Decoding Low-Density Parity-Check Codes With Probabilistic Scheduling," *IEEE Commun. Lett.*, vol. 5, no. 10, pp. 414–416, Oct. 2001.

[6] H. Kfir and I. Kanter, "Parallel Versus Sequential Updating for Belief Propagation Decoding," *Physica A: Statistical Mechanics and its Applications*, vol. 330, no. 1-2, Dec. 2003.

[7] J. Zhang and M. P. C. Fossorier, "Shuffled Belief Propagation Decoding," *Proc. Ann. Asilomar Conf.*, Nov. 2002.

[8] M. M. Mansour and N. R. Shanbhag, "High Throughput LDPC Decoders," *IEEE Trans. VLSI Syst.*, vol. 11, no. 6, pp. 976–996, Dec. 2003.

[9] E. Sharon, S. Lytsin, and J. Goldberger, "An Efficient Message-Passing Schedule for LDPC Decoding," *Proc. 23rd IEEE Convention of Electrical and Electronics Engineers in Israel*, Sept. 2004.

[10] A. I. V. Casado, M. Griot, and R. Wesel, "Informed Dynamic Scheduling for Belief-Propagation Decoding of LDPC Codes," *Proc. IEEE ICC*, Glasgow, Scotland, June 2007.

[11] N. Yacov, H. Efraim, H. Kfir, I. Kanter, and O. Shental, "Parallel vs. Sequential Belief Propagation Decoding of LDPC Codes over GF(q) and Markov Sources," *Physica A: Statistical Mechanics and its Applications*, vol. 378, no. 2, May 2007.

[12] A. Voicila, D. Declercq, F. Verdier, M. Fossorier, and P. Urard, "Low-Complexity Decoding for Non-Binary LDPC Codes in High Order Fields," *IEEE Trans. Commun.*, vol. 58, no. 5, pp. 1365–1375, May 2010.

[13] D. Levin, E. Sharon, and S. Litsyn, "Lazy Scheduling for LDPC Decoding," *IEEE Commun. Lett.*, vol. 11, no. 1, pp. 70–72, Jan. 2007.

[14] H. Song and J. Cruz, "Reduced-Complexity Decoding of Q-ary LDPC Codes for Magnetic Recording," *IEEE Trans. Magn.*, vol. 39, no. 2, pp. 1081–1087, Mar. 2003.

[15] G. J. Byers and F. Takawira, "Fourier Transform Decoding of Non-Binary LDPC Codes," *Proceedings Southern African Telecommunication Networks and Applications Conference (SATNAC)*, Sept. 2004.

[16] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and Irregular Progressive Edge-Growth Tanner Graphs," *IEEE Trans. Inf. Theory*, vol. 51, no. 1, pp. 386–398, Jan. 2005.