# Latency Constrained Protograph-based LDPC Convolutional Codes

Giovanni E. Corazza, Aravind R. Iyengar, Marco Papaleo,
Paul H. Siegel, Alessandro Vanelli-Coralli and Jack K. Wolf

*Invited Paper*

***Abstract*—We propose a windowed decoding scheme for protograph-based LDPC convolutional codes (LDPC-CC) that allows us to efficiently trade-off decoding performance for gains in latency. We study the performance of regular LDPC-CC with the windowed decoding scheme. In particular, we show that the class of LDPC-CC proposed in the literature with good belief propagation performance is ill-suited for windowed decoding. Further, we establish properties of code ensembles with good windowed decoding performance over erasure channels with and without memory.**

## I. Introduction

LDPC Convolutional Codes (LDPC-CC) were first introduced in [3]. These ensembles have several attractive characteristics, such as thresholds approaching capacity with belief-propagation (BP) decoding [4], and BP thresholds close to the maximum a-posteriori (MAP) thresholds of corresponding unstructured ensembles [5]. Whereas irregular LDPC block codes have also been shown to have BP thresholds close to capacity [6], the advantage with convolutional counterparts is that good performance is achieved by relatively simple close-to-regular ensembles. Also, the construction of finite-length codes from LDPC-CC ensembles can be readily optimized to ensure desirable properties, e.g. large girths, fewer cycles, using well-known techniques of LDPC code constructions. Most of these attractive features of LDPC-CC come into effect when they are infinitely long, non-terminated codes. An immediate implication is that BP decoding for non-terminated codes is impractical. However, by implementing a *windowed decoder*, one can get around this limitation of LDPC-CC.

Here, we consider the windowed decoding scheme, introduced in [1], that exploits the convolutional structure of the parity-check matrix of the LDPC-CC to decode non-terminated codes, while maintaining many of the key advantages of iterative decoding schemes like the BP decoder, especially the low complexity and superior performance. When used to decode terminated (block) LDPC-CC, the windowed decoder provides a simple, yet efficient way to trade-off decoding performance for reduced latency. Moreover, the proposed scheme allows for the flexibility to set and change the decoding latency on the fly as required. This proves to be an extremely useful feature when the scheme is used to decode codes over upper layers of the internet protocol.

Here, we review our findings from [7]. We study the requirements of LDPC-CC ensembles for good performance over erasure channels with windowed decoding (WD). We are interested in designing ensembles that have good performances over erasure channels with [2] and without [1] memory with both BP and windowed decoding. Although the channels considered here are erasure channels, we note that the WD scheme can be made use of over any channel. Furthermore, the analysis of the ensemble performance over memoryless erasure channels also applies to the more general class of binary-input, memoryless, output-symmetric (BMS) channels [6].

This paper is organized as follows. Section II introduces protograph-based LDPC convolutional codes. In Section III we describe the windowed decoding algorithm that is based on belief propagation. We briefly discuss the latency reduction and flexibility offered by the windowed decoder. Section IV deals with the asymptotic performance of LDPC-CC ensembles on the binary erasure channel. Moving on to erasure channels with memory, in Section V we analyze LDPC-CC ensembles both in the asymptotic setting and for finite lengths. We note that LDPC-CC ensembles that happen to perform well with the windowed decoding scheme are also the ones that achieve the best performance among all LDPC-CC ensembles over erasure channels with memory. Finally, we conclude in Section VI. For detailed proofs of all results presented in this paper, see [7].

## II. Protograph-based LDPC Convolutional Codes

A protograph [8] is a relatively small bipartite graph from which a larger graph can be obtained by a copy-and-permute procedure – the protograph is copied $M$ times, and then the edges of the individual replicas are permuted among the $M$ replicas to obtain a single, large bipartite graph. Suppose the protograph possesses $N_P$ variable nodes (VNs) and $M_P$ check nodes (CNs). Then the derived graph will consist of $n = N_PM$ VNs and $m = M_PM$ CNs. The nodes of the protograph are labeled, so that if the VN $V_j$ is connected to the CN $C_i$ in the protograph, then $V_j$ in a replica can only connect to one of the $M$ replicated $C_i$'s.

Each protograph can be represented by means of an $M_P \times N_P$ bi-adjacency matrix $\mathbf{B}$, called the *base matrix* of the

A. R. Iyengar, P. H. Siegel and J. K. Wolf are with the Department of Electrical and Computer Engineering and the Center for Magnetic Recording Research, University of California, San Diego, La Jolla, CA 92093 USA (e-mail: aravind@ucsd.edu, psiegel@ucsd.edu, jwolf@ucsd.edu).

The work of A. R. Iyengar is supported by the National Science Foundation under the Grant CCF-0829865.

M. Papaleo , A. Vanelli-Coralli and G. E. Corazza are with the University of Bologna, DEIS-ARCES, Viale Risorgimento, 2 - 40136 Bologna, Italy (e-mail: mpapaleo@arces.unibo.it, avanelli@arces.unibo.it, gecorazza@arces.unibo.it)

Parts of this work were presented at the 2010 IEEE Information Theory Workshop (ITW), Cairo, Egypt [1]; and the 2010 IEEE International Communications Conference (ICC), Cape Town, South Africa [2].



978-1-4244-6746-4/10/$26.00 ©2010 IEEE

protograph where the entry $\mathbf{B}_{i,j}$ represents the number of edges between CN $C_i$ and VN $V_j$ (a non-negative integer, i.e., multiple parallel edges—multiedges—are permitted). The degrees of the VNs (CNs respectively) of the protograph are then equal to the sum of the corresponding column (row, respectively) of $\mathbf{B}$. A $(J, K)$ regular protograph-based code is then one with a base matrix where all VNs have degree $J$ and all CNs, excluding those in the terminated portion of the code, have degree $K$.

The copy-and-permute operation is realized by replacing each edge (multiedge, respectively) in the base matrix $\mathbf{B}$ with a size-$M$ permutation matrix (the sum of as many distinct size-$M$ permutation matrices as the number of multiedges, respectively). The resulting matrix after the above transformation for each element of $\mathbf{B}$ corresponds to the parity-check matrix $\mathbf{H}$ of the code. For different values of $M$, different blocklengths of the derived Tanner graph can be achieved keeping the original graph structure imposed by the protograph. We can hence think of protographs as defining code ensembles that are themselves subsets of unstructured LDPC code ensembles. This means that the density evolution analysis for the ensemble of codes represented by the protograph can be performed within the protograph instead of the unstructured ensemble.

Analogous to LDPC block codes, LDPC-CC can also be derived by a protograph expansion. As for block codes, the parity-check matrices of these convolutional codes are composed of blocks of size-$M$ square matrices.

*1) Classical construction:* We will refer to the construction introduced in [9] as the *classical* construction of $(J, K)$ regular LDPC-CC ensembles and denote these ensembles as $\mathcal{C}_c(J, K)$. Let $a$ be the greatest common divisor (gcd) of $J$ and $K$. Then there exist positive integers $J'$ and $K'$ such that $J = aJ'$, $K = aK'$, and $\gcd(J', K') = 1$. Assuming we terminate the convolutional code after $L$ instants, we obtain a block code, described by the base matrix

$$\mathbf{B} = \overbrace{\begin{pmatrix} \mathbf{B}_0 & & & \\ \mathbf{B}_1 & \mathbf{B}_0 & & \\ \vdots & \mathbf{B}_1 & \ddots & \\ \mathbf{B}_{m_s} & \vdots & \ddots & \mathbf{B}_0 \\ & \mathbf{B}_{m_s} & \ddots & \mathbf{B}_1 \\ & & \ddots & \vdots \\ & & & \mathbf{B}_{m_s} \end{pmatrix}}^{L}$$

where $m_s = a-1$ is the *memory* of the LDPC-CC and $\mathbf{B}_i, i = 0, \cdots, m_s$ are $J' \times K'$ component base matrices that are all identical and have all entries equal to 1. The protograph of the terminated code has $N_P = LK'$ VNs and $M_P = (L + m_s)J'$ CNs. The rate of the LDPC-CC is therefore

$$R_L = 1 - \left(\frac{L + m_s}{L}\right)\frac{J'}{K'} = 1 - \left(1 + \frac{m_s}{L}\right)(1 - R) \quad (1)$$

where $R = 1 - \frac{J'}{K'}$ is the rate of the non-terminated code. Note that $R_L \to R$ and the LDPC-CC has a regular degree distribution [4] when $L \to \infty$.

*2) Modified construction:* We propose a modified construction that is similar to the classical construction except that we do not require that $m_s = a - 1$, i.e. the memory of the LDPC-CC is independent of its degree distribution. We further discard the requirement that the $\mathbf{B}_i$ matrices are identical and have only ones. However, the sizes of the matrices $\mathbf{B}_i$ will still be $J' \times K'$. We will denote a $(J, K)$ regular LDPC-CC ensemble constructed thus as $\mathcal{C}_m(J, K)$. Note that the rate of the $\mathcal{C}_m(J, K)$ ensemble is still given by Equation (1). Further, the independence of the code memory and the degree distribution allows us to construct LDPC-CC even when $J$ and $K$ are co-primes.

## III. Windowed Decoding

LDPC-CC are characterized by a very large value of constraint length $\nu_s = (m_s + 1)K'M$. Since the Viterbi decoder has a complexity that scales exponentially in the constraint length, it is impractical for this kind of code. However, the sparsity of the parity-check matrix can be exploited and an iterative message passing algorithm can be adopted for decoding. Further, the convolutional structure of the code imposes a constraint on the VNs connected to the same parity-check equations—two VNs of the protograph that are at least $(m_s + 1)$ instants apart cannot be involved in the same parity-check equation. This characteristic can be exploited in order to perform continuous decoding of the received stream through a *window* that slides along the bit sequence. In this paper we consider a windowed decoder (WD) suggested by Liva [10] which is mostly reminiscent of the sliding window approach described in [11] for the non-terminated LDPC-CC. Such a decoder is also useful to decode terminated codes to reduce latency.

Consider a terminated $(J, K)$ regular parity-check matrix $\mathbf{H}$ built from a base protograph $\mathbf{B}$. The windowed decoder works on sub-protographs of the code and the window size $W$ is defined as the number of sets of $J'$ CNs of the protograph considered within each window. The window thus consists of $J_W = WJ'M$ rows of $\mathbf{H}$ and all columns that are involved in the check equations corresponding to these rows, i.e., the window covers $WJ'$ rows of the protograph $\mathbf{B}$. It is easy to see that the window size $W$ ranges between $(m_s + 1)$ and $(L - 1)$.

At the first decoding instant, the decoder performs belief-propagation (BP) [12] over the edges within the window with the aim of decoding all of the first $K'M$ symbols in the window, called the *targeted symbols*. The window slides down $J'M$ rows and right $K'M$ columns in $\mathbf{H}$ after a maximum number of BP iterations have been performed, or when all targeted symbols within the window have been successfully recovered; and continues decoding at the new position at the next decoding time instant. We refer to the set of edges included in the window at any particular decoding time instant as the *window configuration*. Since the WD aims to recover only the targeted symbols within each window configuration, the entire codeword is recovered in $L$ decoding time instants. Figure 1 shows a schematic representation of the WD for $W = 4$.

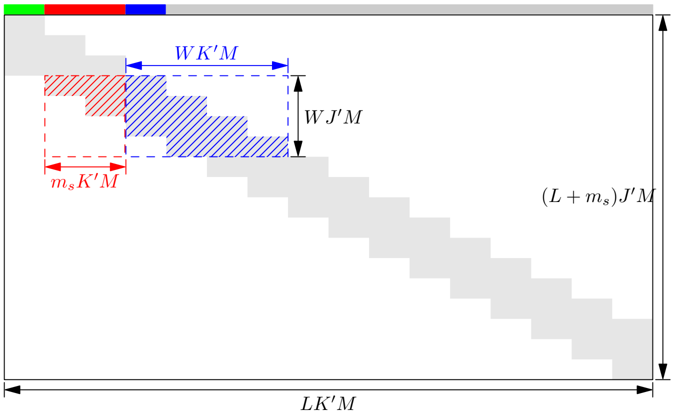


Fig. 1. Illustration of windowed decoding (WD) with window of size $W = 4$ for a $\mathcal{C}_m(J, 2J)$ LDPC-CC with $m_s = 2$ and $L = 16$ at the fourth decoding instant. The window consists of $J_W = WJ'M = 4M$ rows of the parity-check matrix and all the $(W + m_s)K'M = 12M$ columns involved in these equations - this comprises the red and the blue boxes shown. Note that the symbols shown in green above the parity-check matrix have all been processed. Further, if the previous window configurations were all successful in decoding their targeted symbols, the symbols shown in red above the parity-check matrix would also be known, and hence, the window would comprise only the $WK'M = 8M$ columns shown in blue and would have the symbols shown in blue above the parity-check matrix as the targeted symbols. Those symbols whose decoding has not yet been attempted are shown in gray above the parity-check matrix.

The decoding latency of the first $K'M$ symbols with WD is therefore given by

$$\Lambda_{WD} = T_W + T_{dec}(W),$$

where $T_W$ is the time required to receive all the symbols required to decode the first $K'M$ symbols, and $T_{dec}(W)$ is the time taken to decode the targeted symbols. We can relate $T_W$ and $T_{cw}$, the time taken to receive the entire codeword, as

$$T_W = \frac{(W + m_s)K'M}{LK'M} T_{cw} = \frac{W + m_s}{L} T_{cw}.$$

Similarly, $T_{dec}(W)$ and $T_{dec}$, the time taken by the BP deocder working on the entire codeword to decode the first $K'M$ symbols, are related as

$$T_{dec}(W) = \frac{W + m_s}{L} T_{dec},$$

since the complexity of BP decoding scales linearly in block-length. Thus, in latency-limited scenarios, i.e. when waiting for the entire codeword to decode the first few symbols is not affordable, we can use the WD to obtain a latency reduction of

$$\Lambda_{WD} = \frac{W + m_s}{L} \Lambda_{BP}.$$

For the sequence of ensembles indexed by $L$, with the choice of the proposed WD with a fixed finite window size $W$, the latency vanishes as $1/L$.

*Latency Flexibility:* Although reduced latency is an important characteristic of WD, what is perhaps more useful practically is the flexibility to alter the latency with corresponding changes in the code performance. At the decoder, the latency can be controlled by varying the parameter $W$ as required. If a large latency can be handled, $W$ can be kept large ensuring good code performance and if a small latency is required, $W$ can be made small while paying a price in the code performance. For the same code at the encoder, different latencies can be achieved with different error rates, and the latencies can be adjusted on the fly.

## IV. Memoryless Erasure Channels

In this section, we confine our attention to the performance of the LDPC-CC when the transmission occurs over a memoryless erasure channel, i.e. a binary erasure channel (BEC). Further, we consider the performance of the LDPC-CC in terms of the average performance of the codes belonging to ensembles defined by protographs in the limit of infinite block-lengths (when the protographs are expanded using infinite-sized square matrices) and in the limit of infinite iterations of the decoder. We will therefore concentrate on the erasure-rate *thresholds* [6] of the code ensembles as a performance metric in our search for good LDPC-CC ensembles.

The asymptotic analysis for WD is very similar to that of the BP decoder owing to the fact that the part of the code within a window is itself a protograph-based code. However, here the decoding within a window is successful as long as the probability of failing to recover the targeted symbols goes to zero. We define the threshold $\varepsilon^*_{(i)}$ of the $i^{\text{th}}$ window configuration as the supremum of the channel erasure-rates for which the WD succeeds in retrieving the targeted symbols of the $i^{\text{th}}$ window given all targeted symbols corresponding to the first $(i-1)$ window configurations. The *windowed threshold* $\varepsilon^*_W$ of a code ensemble with a decoder of window size $W$ is then defined as the supremum of the set of channel erasure-rates for which WD is successful for all window configurations.

We assume that between decoding time instants, no information is carried forward, i.e., when a particular window configuration has been processed, all the present processing information apart from the decoded targeted symbols themselves is discarded. With this assumption, the only dependence between two successive window configurations is through the decoded symbols. We will refer to this characteristic of the proposed decoder as *inter-window independence*. This assumption gives us the following.

*Proposition 1: The windowed threshold of an LDPC-CC ensemble is the minimum of the thresholds of its window configurations.*

Note that for non-terminated LDPC-CC, the above claim would become

$$\varepsilon^*_W = \inf_i \{\varepsilon^*_{(i)}\}.$$

*Corollary 2: For a $\mathcal{C}_m(J, K)$ ensemble, $\varepsilon^*_W = \varepsilon^*_{(1)}$.*

We can therefore concentrate on the performance of the first window configuration of LDPC-CC ensembles.

*Window rate:* The sub-protograph within the first window has a rate $R_W$ independent of the termination length $L$ of the protograph for $m_s + 1 \leq W \leq L$. We can hence talk of the "Shannon limit for an ensemble for WD". For a $\mathcal{C}_m(J, K)$ LDPC-CC ensemble, the rate of the code within the window of size $W$ satisfies therefore $R_W \leq R$.

The following monotonicity result can be shown based on the monotonicity of the BP decoding rules.

*Proposition 3:* $\varepsilon^*_{W+1} \geq \varepsilon^*_W$.

We will say an LDPC-CC ensemble is "bad for WD" if $\varepsilon^*_W = 0$ for all valid window sizes $W$. The following proposition shows that classical constructions of LDPC-CC are ill-suited for WD.

*Proposition 4:* $\varepsilon^*_W = 0 \, \forall \, m_s + 1 \leq W \leq L$ *for the ensemble* $\mathcal{C}_c(J, K'J)$.

The classical $(3, 6)$ regular LDPC-CC ensemble $\mathcal{C}_1$ given by $\mathbf{B}_0 = \mathbf{B}_1 = \mathbf{B}_2 = [1 \quad 1]$ thus has $\varepsilon^*_W = 0$ although it has a BP threshold $\varepsilon^*_{BP} \approx 0.4881$ [9]. Modified constructions are more amenable to WD. However, these ensembles need to be carefully constructed to avoid bad structures within the protographs:

*Proposition 5: If* $\exists \, j \in [K']$ *such that* $\mathbf{B}_{0_{1,j}} = 1$*, i.e. if the* $\mathbf{B}_0$ *matrix contains a* $1$*, then the* $\mathcal{C}_m(J, K'J)$ *ensemble is bad for WD.*

While the above proposition provides a necessary condition for a $\mathcal{C}_m(J, K'J)$ ensemble to have a positive windowed threshold, it does not guarantee it. The windowed threshold of an ensemble is, in general, numerically evaluated using the P-EXIT analysis in [13] with suitable modifications to track the convergence of the a-posteriori mutual information at the targeted symbols in the sub-protograph representing the window. Owing to the fact that the P-EXIT computations are performed numerically, the threshold estimates obtained are *conjectured* to be the actual thresholds. The similarity of the convergence behavior of the P-EXIT computations to that of randomized LDPC ensembles and the observation of the behavior of finite-length codes through Monte-Carlo simulations constitute evidences in favor of the conjecture being true.

In Figure 2, we show the windowed thresholds plotted against the window size for two $\mathcal{C}_m(3, 6)$ ensembles—ensemble $\mathcal{C}_2$ of memory $m_s = 3$ (same as that of $\mathcal{C}_1$) specified by $\mathbf{B}_0 = [2 \quad 2], \mathbf{B}_1 = [0 \quad 1], \mathbf{B}_2 = [1 \quad 0]$, and ensemble $\mathcal{C}_3$ of memory $m_s = 2$ specified by $\mathbf{B}_0 = [2 \quad 2], \mathbf{B}_1 = [1 \quad 1]$. Observe that the monotonicity result of Proposition 3 is apparent. Also note that the threshold is fairly close to the maximum windowed threshold even when $W = m_s + 1$, and the maximum windowed threshold is itself fairly close to the BP threshold $\varepsilon^*_{BP}$.

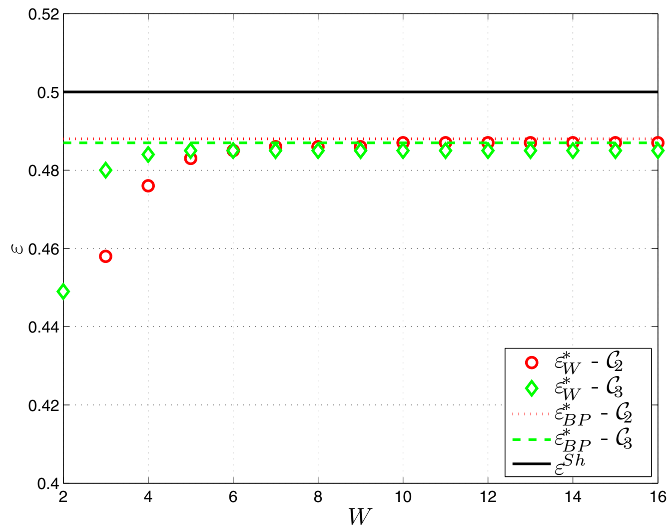


Fig. 2. Windowed threshold as a function of the window size for the ensembles $\mathcal{C}_2$ and $\mathcal{C}_3$ with $L = 20$. Also shown are the Shannon limits for WD for the two ensembles and the corresponding BP thresholds. The window rate of both codes $R_W$ is the same as the non-terminated code rate $R = 0.5$ whereas the rates of the terminated codes are 0.45 and 0.475 for ensembles $\mathcal{C}_2$ and $\mathcal{C}_3$ respectively.

## V. Erasure Channels with Memory

The performance metric of a code over a bursty erasure channel is related to the maximum resolvable erasure burst length (MBL) denoted $\Delta_{max}$ [14]; which, as the name suggests, is the maximal length of a single solid erasure burst that can be decoded. Methods of optimizing codes for such channels therefore focus on permuting columns of parity-check matrices to maximize $\Delta_{max}$, e.g. [15]–[20]. Instead of permuting columns of the parity-check matrix, in order to maintain the convolutional structure of the code, we will consider designing $\mathcal{C}_m(J, K)$ ensembles that maximize $\Delta_{max}$.

When the BP decoder is used to decode a terminated LDPC-CC over an erasure channel with memory, the performance depends on stopping sets of the code. The structure of protographs imposes constraints on the code that limit the stopping set sizes and locations. Let us define a *protograph stopping set* to be a subset $S(\mathbf{B})$ of the VNs of the protograph $\mathbf{B}$ whose neighboring CNs are connected at least twice to $S(\mathbf{B})$. We call the least number of consecutive columns of $\mathbf{B}$ that contain the stopping set $S(\mathbf{B})$ as the *span* of the stopping set, denoted $\langle S(\mathbf{B})\rangle$. Let us denote the minimum number of consecutive columns of the protograph $\mathbf{B}$ that contains a protograph stopping set by $\langle S(\mathbf{B})\rangle^*$. When the protograph under consideration is clear from the context, we will drop it from the notation and use $\langle S\rangle^*$. The minimum span of a stopping set is of interest because we can give simple bounds for $\Delta_{max}$ based on $\langle S(\mathbf{B})\rangle^*$. Our aim in the following will be to obtain bounds for the maximal $\langle S(\mathbf{B})\rangle^*$ over $\mathcal{C}_m(J, K)$ ensembles with memory $m_s$, which we denote $\langle S(J, K, m_s)\rangle^*$, and design protographs that achieve minimal spans close to this optimal value.

Observing that if one of the VNs in the protograph is connected multiply to all its neighboring CNs, then it forms a protograph stopping set by itself. In order to obtain a larger minimum span of stopping sets, it is desirable to avoid this case. The following can be shown for $(J, 2J)$ regular ensembles.

*Proposition 6: For any* $\mathcal{C}_m(J, 2J)$ *ensemble with memory* $m_s$*,* $\langle S\rangle^* \leq 2m_s$.

*Proposition 7: For* $\mathcal{C}_m(J, 2J)$ *ensembles with memory* $m_s$ *and* $J > 2$*,* $\langle S(J, 2J, m_s)\rangle^* = 2m_s$.

Note that the ensemble $\mathcal{C}_2$, with $\langle S\rangle^* = 4$, achieves the optimal span in Proposition 7. In this sense, the modified construction allows us to construct the best possible LDPC-CC ensembles. We bring to the reader's attention here that the optimal span achieving constructions are not unique in general. These constructions allow us to design $\mathcal{C}_m(J, 2J)$

ensembles for a wide range of required $\langle S\rangle^*$. A drawback of the convolutional structure, however, is that if $m_s$ is increased to get larger $\langle S\rangle^*$, the code rate $R_L$ decreases linearly for a fixed $L$. We can show that the $\mathcal{C}_m(J,K)$ ensembles perform at least as well as the corresponding $\mathcal{C}_m(a,K)$ ensembles.

*Proposition 8:* $\langle S(J,K,m_s)\rangle^* \geq \langle S(a,K,m_s)\rangle^*$ *where* $a = \gcd(J,K) \geq 2$.

In fact, it is usually possible to strictly improve the minimal span of the $\mathcal{C}_m(J,K)$ ensemble over the $\mathcal{C}_m(a,K)$ ensemble.

The asymptotic analysis for WD is essentially the same as that for BP. We are now interested in the sub-protograph stopping sets, denoted $S_W$, that include the targeted symbols within a window. Let us denote the minimal span of such stopping sets as $\langle S_W\rangle^*$. Since stopping sets of the protograph of the LDPC-CC are also stopping sets of the sub-protograph within a window, and since such stopping sets can be made to include the targeted symbols within the window, we have $\langle S_W\rangle^* \leq \langle S\rangle^*$. Further it can easily be shown that the minimum stopping set span within the window is monotonic in window size

$$\langle S_{W+1}\rangle^* \geq \langle S_W\rangle^*.$$

We now show the relation between the parameters $\Delta_{max}$ and $\langle S(\mathbf{B})\rangle^*$. We shall assume in the following that $\langle S\rangle^* \geq 2$. We will assume that the protographs are expanded by a factor $M$ to obtain codes.

*Proposition 9: For any* $(J,K)$ *regular LDPC-CC,* $\Delta_{max} \leq M\langle S\rangle^* - 1$.

From the above result, it can be shown that the MBL of regular LDPC-CC is bounded away from that of MDS codes of the same rate and blocklength in general. Despite this discouraging result, we can guarantee an MBL that linearly increases with $\langle S\rangle^*$:

*Proposition 10: For any* $(J,K)$ *regular LDPC-CC,* $\Delta_{max} \geq M(\langle S\rangle^* - 2) + 1$.

The MBL for WD, $\Delta_{max}(W)$, can be bounded as in the case of BP based on $\langle S_W\rangle^*$. Propositions 9 and 10 in this case imply that

$$M(\langle S_W\rangle^* - 2) + 1 \leq \Delta_{max}(W) \leq M\langle S_W\rangle^* - 1.$$

## VI. Conclusions

We considered a windowed decoding scheme for LDPC convolutional codes that allows for the decoding of non-terminated codes. When used to decode terminated LDPC-CC, this scheme allows the efficient trade-off of decoding performance for reduced latency. The asymptotic performance of LDPC-CC ensembles with the proposed WD was analysed, both for the BEC and for erasure channels with memory. Bad structures within protographs of classical LDPC-CC were avoided in a modified construction, which allowed for the construction of ensembles that performed well with WD. Over erasure channels with memory, a class of LDPC-CC ensembles with the modified construction were shown to be best among all the LDPC-CC ensembles. Whereas close-to-optimal performance (in the sense of approaching capacity) was achievable for the BEC, the structure of LDPC-CC imposed constraints that bounded the performance over erasure channels with memory strictly away from the optimal performance (in the sense of approaching MDS performance). Nevertheless, the simple structure and good performance of these codes, as well as the latency flexibility and low complexity of the decoding algorithm, are attractive characteristics for practical systems.

## Acknowledgement

The authors would like to thank Gianluigi Liva who suggested the trade-off of decoding performance for reduced latency through the use of a windowed decoder.

## References

[1] M. Papaleo, A. R. Iyengar, P. H. Siegel, J. Wolf, and G. Corazza, "Windowed erasure decoding of LDPC convolutional codes," in *2010 IEEE Information Theory Workshop*, Cairo, Egypt, Jan. 2010, pp. 78–82.
[2] A. R. Iyengar, M. Papaleo, G. Liva, P. H. Siegel, J. K. Wolf, and G. E. Corazza, "Protograph-based LDPC convolutional codes for correlated erasure channels," in *IEEE Int'l Conf. Comm.*, Cape Town, South Africa, May 2010.
[3] A. J. Felstrom and K. Zigangirov, "Time-varying periodic convolutional codes with low-density parity-check matrix," *IEEE Trans. Info. Theory*, vol. 45, no. 6, pp. 2181–2191, Sep. 1999.
[4] M. Lentmaier, G. P. Fettweis, K. S. Zigangirov, and J. D. J. Costello, "Approaching capacity with asymptotically regular LDPC codes," in *Information Theory and Applications*, San Diego, California, 2009.
[5] S. Kudekar, T. Richardson, and R. L. Urbanke, "Threshold saturation via spatial coupling: Why convolutional LDPC ensembles perform so well over the BEC," *CoRR*, vol. abs/1001.1826, 2010.
[6] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008.
[7] A. R. Iyengar, M. Papaleo, P. H. Siegel, J. K. Wolf, A. Vanelli-Coralli, and G. E. Corazza, "Windowed decoding of protograph-based LDPC convolutional codes over erasure channels," *IEEE Trans. Info. Theory (Submitted)*, 2010.
[8] J. Thorpe, "Low-density parity-check (LDPC) codes constructed from protographs," JPL INP, Tech. Rep., Tech. Rep., Aug. 2003.
[9] A. Sridharan, D. V. Truhachev, M. Lentmaier, J. D. J. Costello, and K. S. Zigangirov, "Distance bounds for an ensemble of LDPC convolutional codes," *IEEE Trans. Info. Theory*, vol. 53, no. 12, pp. 4537–4555, Dec. 2007.
[10] G. Liva, private communication, 2008.
[11] A. Pusane, A. J. Feltstrom, A. Sridharan, M. Lentmaier, K. Zigangirov, and D. Costello, "Implementation aspects of LDPC convolutional codes," *IEEE Trans. Commun.*, vol. 56, no. 7, pp. 1060–1069, Jul. 2008.
[12] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Info. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.
[13] G. Liva and M. Chiani, "Protograph LDPC codes design based on EXIT analysis," in *Proc. IEEE Globecom*, Washington, D.C., 2007, pp. 3250–3254.
[14] M. Yang and W. E. Ryan, "Design of LDPC codes for two-state fading channel models," in *The 5th International Symposium on Wireless Personal Multimedia Communications*, vol. 3, Oct. 2002, pp. 986–990.
[15] S. J. Johnson and T. Pollock, "LDPC codes for the classic bursty channel," in *Proc. IEEE Int'l Symp. Info. Theory*, Aug. 2004, pp. 184–189.
[16] E. Paolini and M. Chiani, "Improved low-denisty parity-check codes for burst erasure channels," in *IEEE Int'l Conf. Comm.*, Istanbul, 2006.
[17] G. Liva, B. Matuz, Z. Katona, E. Paolini, and M. Chiani, "On construction of moderate-length LDPC codes over correlated erasure channels," in *IEEE Int'l Conf. Comm.*, Dresden, 2009.
[18] G. Sridharan, A. Kumarasubramanian, A. Thangaraj, and S. Bhashyam, "Optimizing burst erasure correction of LDPC codes by interleaving," in *Proc. IEEE Int'l Symp. Info. Theory*, Jul. 2008, pp. 1143–1147.
[19] S. J. Johnson, "Burst erasure correcting LDPC codes," *IEEE Trans. Commun.*, vol. 57, no. 3, pp. 641–652, Mar. 2009.
[20] E. Paolini and M. Chiani, "Construction of near-optimum burst erasure correcting low-density parity-check codes," *IEEE Trans. Commun.*, vol. 57, no. 5, pp. 1320–1328, May 2009.