

Rate-Compatible Punctured Low-Density Parity-Check Codes for Ultra Wide Band Systems

Demijan Klinc^{*}, Jeongseok Ha[†], Jaehong Kim[‡] and Steven W. McLaughlin[‡]

^{*}Institute for Communications Engineering (LNT)

Technische Universität München, 80290 München, Germany

Email: demi@gimb.org

[†]Information and Communications University

103-6 Munji-dong Yuseong-gu, Daejeon 305-732, South Korea

Email: jsha@icu.ac.kr

[‡]School of Electrical and Computer Engineering

Georgia Institute of Technology, Atlanta, Georgia 30332-0250

Email: onil@ece.gatech.edu and swm@ece.gatech.edu

Abstract—In this paper¹ we treat rate-compatible punctured low-density parity-check (RCP-LDPC) codes. We show how a mother code can be efficiently punctured to achieve good performance over a wide range of rates and discuss the design of a mother code. While this approach can be applied to virtually any time-varying channel, we apply it to the ultra wide band (UWB) channel, e.g. 802.15.3a, and we show that RCP-LDPC codes perform well as compared to a selection of fixed rate unpunctured codes. They reduce the implementation complexity and enable the use of retransmission protocols based on incremental redundancy to increase the throughput.

I. INTRODUCTION

Modern communication systems operating over time variant channels must be capable of adapting the rate according to the available channel state information in order to guarantee the desired system performance at all times and maximize the throughput. Conventional applications usually employ a selection of fixed rate coding schemes, where each of them is optimized independently for the given rate to meet the desired performance. Unfortunately, the complexity of such systems is rather high, since usually a separate encoder/decoder pair is required for each coding scheme.

An alternate way of realizing rate adaptability is to systematically puncture a low rate code, a so-called *mother code*, such that the bits of higher rate codes are embedded in those of lower rate codes [1]. Such a set of codes is called *rate-compatible*. The beauty of rate-compatible codes lies in the fact that only one encoder/decoder pair is needed for the entire range of rates, given the puncturing locations are known to the receiver. Furthermore, rate-compatible codes enable the use of retransmission protocols based on incremental redundancy such as the type-II hybrid Automatic Repeat Request (ARQ) protocol, which can increase the overall system throughput in case of channel impairments. However, the code design at different rates is limited due to the rate-compatibility constraint.

Our objective is to show how rate-compatible punctured low-density parity-check (RCP-LDPC) codes can be designed for use in future digital wireless data transmission systems that operate in time-varying environments and require high data-rates. A good example of such systems is the upcoming Ultra Wide Band (UWB) technology that is currently still in the process of standardization under IEEE. We design RCP-LDPC codes for the use in UWB systems and address some important issues that need to be considered.

Low-density parity-check (LDPC) codes are defined as linear block codes with a very sparse parity-check matrix [2]. Very often the code is described in terms of a Tanner graph [3] that consists of variable nodes, check nodes and edges between them — see Figure 1. The variable and check nodes represent

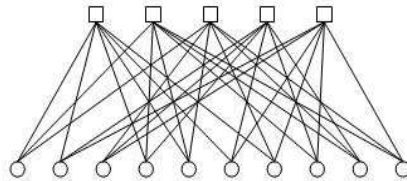


Fig. 1. The Tanner graph. Variable nodes, representing the codeword bits, are depicted as circles and check nodes, representing the parity-check equations, are depicted as squares.

the codeword bits and the parity-check constraints, respectively, while the edges define the dependencies between the two. Say, if a variable node i is connected with a check node j , the i -th codeword bit is involved in the j -th parity check constraint. The number of edges emanating from a node defines its degree.

An ensemble of LDPC codes is defined with a degree distribution pair

$$\lambda(x) = \sum_{i=2}^{d_v} \lambda_i x^{i-1} \quad (1)$$

$$\rho(x) = \sum_{i=2}^{d_c} \rho_i x^{i-1}, \quad (2)$$

¹This work was funded by Samsung Advanced Institute of Technology.

where λ_i is the fraction of edges emanating from variable nodes of degree i , ρ_i is the fraction of edges emanating from check nodes of degree i , and d_v and d_c denote the maximum variable node and check node degrees, respectively. There are many possible Tanner graphs that conform to a particular degree distribution pair, therefore we say that a Tanner graph is an instance of the ensemble.

Effective decoding algorithms yield remarkable performance of LDPC codes that can closely approach the theoretical bounds. During the decoding process messages are iteratively exchanged between variable and check nodes until all parity-check constraints are fulfilled or the maximum number of iterations has been reached. In the latter case, the decoder reports an error since it was unable to find the transmitted codeword. If the decoder finds the combination of bits that satisfy all parity-check constraints it terminates with success and usually the decoded codeword also equals the transmitted one. However, if an LDPC code is not designed carefully the occurrence of the decoded codeword not equaling the transmitted one, which we refer to as *decoder error*, becomes more frequent and additional data-reliability techniques such as cyclic redundancy check (CRC) have to be added.

The following sections are organized as follows. In the second section we briefly describe how bits of the LDPC mother code are punctured to achieve higher rates. In the third section we address the problem of designing a good mother code and subsequently, in the fourth section, we show how RCP-LDPC codes perform over a UWB channel. In the end, we give a summary of the conducted work.

II. THE PUNCTURING ALGORITHM

Puncturing of the mother code can be performed either randomly or according to intelligently chosen puncturing locations, optimized for a given parity-check matrix or Tanner graph. By smartly choosing sequences of bits to be punctured, we can significantly improve the code's performance at a wide range of rates as compared to random puncturing.

Suppose that some bits in the transmitted codeword were punctured by the encoder and hence not transmitted. Assuming a uniform source, the decoder will set the log-likelihood ratio (LLR) values of the variable nodes that represent punctured bits to 0, since no a-priori information about them is provided. If during the decoding process such a variable node receives a non-zero LLR message from one of its neighboring check nodes, it is said to be *recovered*, regardless of whether the LLR value implies the correct bit value or not. Through iterations, all "punctured" variable nodes have to be recovered with the information provided by the unpunctured bits, otherwise the parity-check constraints can never be satisfied.

A punctured variable node is called *one step recoverable (1-SR)* if it has at least one check node neighbor (called survived check node whose remaining variable node neighbors are all unpunctured (see Figure 2). We note that a 1-SR variable node is guaranteed to be recovered after the first iteration.

In a similar way, a k -SR variable node can be defined [4][5], which is recovered after the k -th iteration. It can be shown

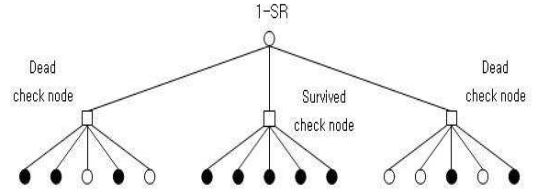


Fig. 2. A 1-SR variable node. Void and filled circles are punctured and unpunctured variable nodes, respectively.

[4] that the probability of a punctured variable node being recovered with a wrong value usually rises with the number of iterations required for the recovery. Thus the mother code should be systematically punctured such that the punctured bits are recovered in as few iterations as possible. For choosing which variable node should be punctured and which should stay unpunctured we propose an algorithm in [4]. The algorithm first attempts to maximize the number of 1-SR nodes. When no more 1-SR nodes can be found, it proceeds with 2-SR nodes, etc., until every variable node (bit) has been chosen to be either punctured or unpunctured.

Say that all punctured variable nodes are recovered after K or less iterations. Then the variable nodes can be organized in groups G_0, G_1, \dots, G_K , where the index of the group indicates the level of recoverability of variable nodes in it and G_0 contains unpunctured variable nodes. The maximum achievable rate R_{\max} can be expressed as

$$R_{\max} = \frac{R_0}{1 - \sum_{j=1}^K |G_j|/N}, \quad (3)$$

where R_0 is the rate of the mother code and N is the block length. For a sequence of desired rates, $R_0 \leq R_1 \leq \dots \leq R_{\max}$, let P_i be the set of variable nodes that are punctured to achieve rate R_i . The required number of variable nodes in P_i is

$$|P_i| = \left\lceil \frac{N(R_i - R_0)}{R_i} \right\rceil. \quad (4)$$

To achieve R_{\max} we simply puncture all variable nodes as determined by the grouping algorithm, thus $P_{\max} = G_1 \cup \dots \cup G_K$. However, to achieve rates between R_0 and R_{\max} only a fraction of the nodes from P_{\max} has to be punctured. Moreover, P_i must be a subset of P_j if $R_i < R_j$, since the codes must be rate-compatible. The best performance is achieved when variable nodes from lower indexed groups are punctured first.

The rate-compatible puncturing as we propose yields better performance as compared to random puncturing, whereby the performance gap between them increases with the rate. Also, some selections of punctured variable nodes are prevented that lead to catastrophic behavior of the code, which happens when some punctured variable nodes can not be recovered during the decoding process. For additional information together with the in-depth analysis of the algorithm we refer to [4].

III. DESIGN OF A MOTHER CODE

Another essential issue is the design of a mother code. If a mother code is designed carefully, the performance of RCP-LDPC codes can be further improved over the entire range of rates. Towards this end, two things are of vital importance: optimization of the mother code in terms of its performance and resilience of the mother code against the introduced puncturing.

As we mentioned earlier, the decoder does not possess any information about the punctured bits and sets their LLR value to 0. These bits can hence be regarded as erased during transmission. In addition, broadband wireless systems increasingly use multi-carrier modulation, like Orthogonal Frequency Division Multiplex (OFDM). Usually in such systems the coherence bandwidth of the transmission channel is much smaller than the system bandwidth, therefore many spectral nulls are expected in the channel frequency response. Consequently, data transmitted at frequencies that are strongly attenuated on the channel can also be regarded as erased, since the receiver will practically have no information about them.

With these facts in mind, we choose to design the mother code over the binary erasure channel. We search for a code that can withstand the highest erasure probability of the channel (called *threshold*) in the sense that the decoder is able to recover the erased/punctured bits and decode the codeword. If we denote the probability that a message from an arbitrary variable node in the Tanner graph of the mother code sent along an adjacent edge in the k -th iteration is zero (in the log-likelihood domain) as $e^{(k)}$,

$$e^{(k)} \rightarrow 0, \quad \text{as } k \rightarrow \infty \quad (5)$$

has to be fulfilled. The channel erasure probability for which (5) is true has to be maximized in order to increase the resilience of the mother code against erasures and thus against puncturing. Alternatively, if we denote R_0 as the rate of the mother code, we can maximize the function

$$f(e) = \frac{R_0}{1-e}, \quad (6)$$

which can also be interpreted as the highest achievable rate of the punctured code. According to [6], $e^{(k)}$ can be expressed with the following recursive formula

$$e^{(k)} = e^{(0)} \lambda \left(1 - \rho \left(1 - e^{(k-1)} \right) \right), \quad (7)$$

where the functions $\lambda(x)$ and $\rho(x)$ are defined by (1) and (2), respectively, and $e^{(0)}$ equals the erasure probability of the channel. From here, the degree distribution that allows the maximum erasure probability of the channel can be obtained by means of linear programming.

This method produces an optimal degree distribution in the asymptotical sense — that is in the limit when the block length goes to infinity. In practice very often the block length can not be very long due to severe delay and implementation limitations. For codes with short block lengths additional issues have to be addressed and we address them on an

example by designing a mother code with a block length of 2400.

It is known that variable nodes with degree two nodes can increase the threshold at the cost of decreasing the code's minimal distance which causes higher error floors. If conventional random parity-check matrix construction is applied at such a short block length, the inclusion of variable nodes with degree two can cause the code to have unacceptably high error floors. Nevertheless, by using some advanced construction techniques, such as the progressive edge growth (PEG) algorithm [7], the minimal distance can be improved considerably and error floor suppressed out of the usual target bit error rate (BER) of 10^{-5} . The question is how big the fraction of degree two variable nodes, λ'_2 , should be. The higher value of λ'_2 increases the threshold, but a big fraction of degree two variable nodes can result in a small minimum distance, causing a greater probability of decoder errors. We tried to find the optimal λ'_2 by varying it from 0.30 to 0.50 in increments of 0.05. The results are summarized in Table I.

Code	λ'_2	e	λ_2	λ_3	λ_8	ρ_6	ρ_7
GT-1	0.300	0.458	0.189	0.569	0.242	0.6	0.4
GT-2	0.350	0.462	0.220	0.513	0.267	0.6	0.4
GT-3	0.400	0.467	0.251	0.456	0.293	0.6	0.4
GT-4	0.450	0.472	0.283	0.399	0.318	0.6	0.4
GT-5	0.500	0.477	0.319	0.357	0.324	0.6	0.4

TABLE I
PARAMETERS OF THE DESIGNED CODES

The mother code with a block length of 2400 was constructed for each of the degree distributions from Table I with the PEG algorithm and subsequently 800 bits were punctured according to the puncturing algorithm from [4], to achieve the rate 0.75. The performance over the AWGN channel in terms of the BER with respect to E_b/N_0 is shown in Figure 3, where the maximum number of iterations was set to 100.

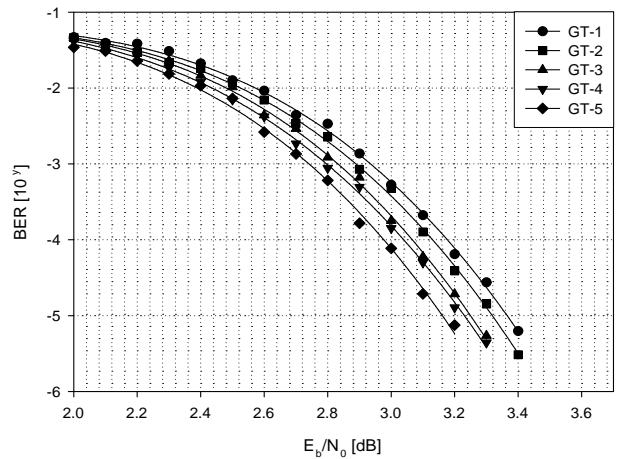


Fig. 3. The BER performance of the punctured codes from Table I at $R = 0.75$.

The performance evidently improves with the higher λ'_2 . At the target BER of 10^{-5} , the GT-5 code, where $\lambda'_2 = 0.50$,

exhibits a gap of 0.2 dB over the GT-1 code, where $\lambda'_2 = 0.30$. Therefore, according to Figure 3 a clear winner would be GT-5. However, we mentioned earlier that with growing λ'_2 the minimum distance is expected to be smaller which causes a higher probability of a decoder error. In case of the GT-5 code, various decoder errors were observed, while none were observed with the other four codes. In order to avoid the need for additional data reliability techniques we choose GT-4 as the best code where no decoder errors could be observed.

IV. RCP-LDPC CODES FOR THE UWB STANDARD

In this section we show how RCP-LDPC codes can be applied in the upcoming UWB standard, which we regard as a good example of future digital wireless data transmission systems. We discuss the performance of RCP-LDPC codes over the UWB channel as compared to another proposed coding scheme and address some further issues that have to be considered in the design process.

The specification of the physical layer for the UWB standard (IEEE 802.15.3a) is still under development. There are two proposals: one based on the Multiband Orthogonal Frequency Division Multiplex (MB-OFDM) and another based on Direct Sequence Code Division Multiple Access (DS-CDMA). All simulations in this section are based on the MB-OFDM proposal [8].

The available bandwidth, ranging from 3.1 GHz to 10.6 GHz, is divided into several 528 MHz bands, where information is transmitted using OFDM modulation on each band². Current proposal incorporates a wide range of possible data rates from 53.3 Mbit/s to 480 Mbit/s. They are achieved by changing the rate of the channel coding scheme and by using time- and frequency-domain spreading. Two channel coding schemes are considered as candidates: rate-compatible punctured convolutional codes and LDPC codes – the convolutional codes are considered as the prime candidate, while efforts are being made to employ LDPC codes as an optional coding scheme. Information about the UWB channel model can be found in [9].

As a reference in our simulations, which were performed according to the block diagram in Figure 4, we use one of the channel coding proposals that uses unpunctured LDPC codes with block length 2400 at three different rates: 0.5, 0.625, and 0.75. We refer to these codes as reference codes and their parameters are summarized in Table II.

Rate	$\lambda(x)$	$\rho(x)$
0.5	$0.771x^2 + 0.229x^7$	x^6
0.625	$0.680x^2 + 0.320x^7$	x^9
0.75	$0.753x^2 + 0.247x^7$	$0.797x^{13} + 0.203x^{14}$

TABLE II
PARAMETERS OF THE REFERENCE CODES.

On the other hand, we use the mother code, GT-4, that we designed in section III and puncture it to achieve rates

²There are 128 carriers in one OFDM symbol.

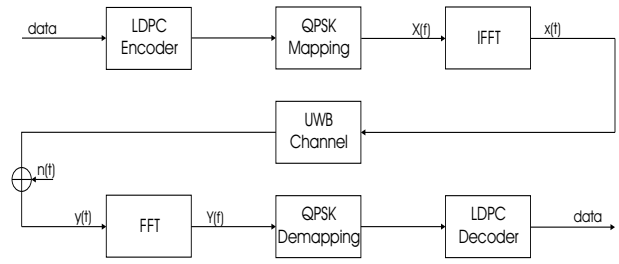


Fig. 4. The UWB simulator block diagram. The transmitted and the received signal are denoted by $x(t)$ and $y(t)$, respectively, while $n(t)$ denotes AWGN which is superimposed to the received signal.

0.625 and 0.75. The performance of these rate-compatible punctured codes has to be evaluated over the UWB channel and compared with the reference codes.

Especially critical is the performance gap with respect to the reference code at the highest rate of 0.75. In simulations in section III the maximum number of iterations was set to 100. So many iterations will very likely be unfeasible in a real-life UWB system, therefore we should consider setting an appropriate maximum number of iterations that would yield a good trade off between the performance and the implementation complexity.

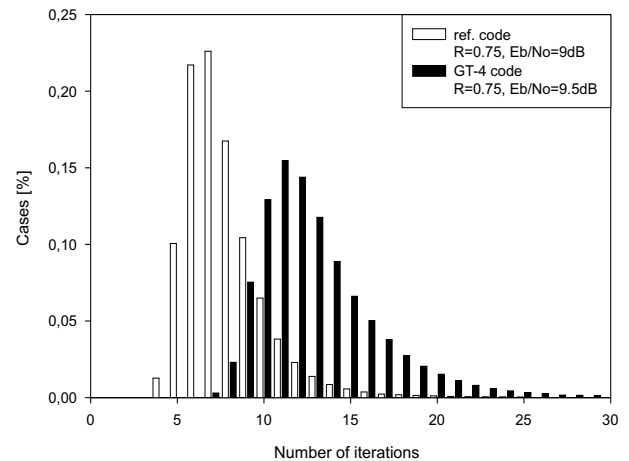


Fig. 5. The distribution of required number of iterations over the UWB channel at $\text{BER} = 10^{-5}$.

Towards this end, we set the maximum number of iterations once again to 100 and observe how many iterations were required to successfully decode a transmitted codeword with the rate 0.75 when the BER of the system was 10^{-5} . Figure 5 shows the results.

Clearly, the punctured GT-4 code requires approximately 10 more iterations to decode successfully than the unpunctured reference code, which is somewhat intuitive since the recovery of the punctured bits requires a few iterations. The maximum number of iterations for the reference codes was set to 12, which seems reasonable according to Figure 5. Raising that number would not result in considerable improvement in performance. Namely, in the vast majority of cases 12 iterations

are sufficient for decoding a transmitted codeword and more iterations would only result in a small performance gain.

In contrast, the maximum number of iterations for the punctured GT-4 code has to be set considerably higher. Figure 6 shows the performance of the punctured GT-4 code at $R = 0.75$ with different maximum number of iterations. A

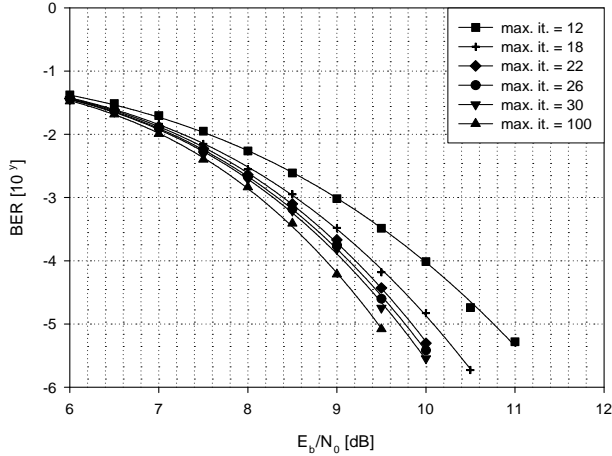


Fig. 6. The BER performance of the punctured GT-4 at $R = 0.75$ with different maximum number of iterations.

significant improvement in performance can be achieved by increasing the maximum number of iterations to 22. Raising that number to 26 results in a very small performance gain of 0.05 dB, while the complexity can increase considerably. It seems that setting the maximum number of iterations to 22 is a good compromise.

Now, we can compare our RCP-LDPC codes (GT-4) with the reference codes over the UWB channel at all considered rates. The results are shown in Figure 7. At the lowest rate of

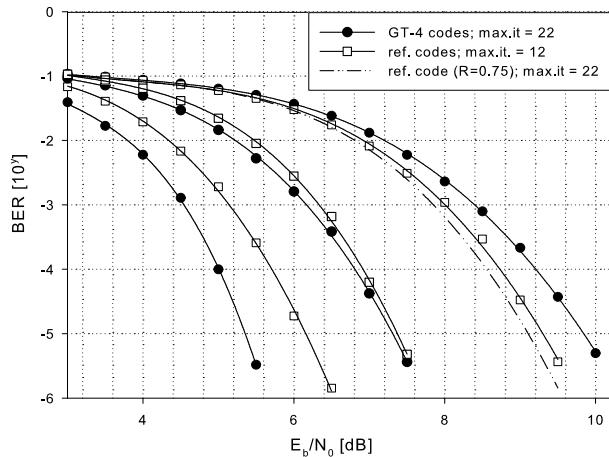


Fig. 7. The BER performance comparison between the reference and GT-4 codes. The curves from left to right correspond to rates 0.5, 0.625, and 0.75.

0.5 the GT-4 mother code outperforms the reference code with a margin of 0.8 dB at the BER of 10^{-5} . The block length of both codes is 2400. On the other hand, at rates 0.625 and 0.75

the GT-4 codes are punctured and have accordingly shorter block lengths (1920 and 1600), while the block length of the reference codes stays 2400. Therefore, a direct comparison is not fair from the information theoretical point of view. Nevertheless, at the rate 0.625 the performance difference between the punctured GT-4 code and the reference code is almost indistinguishable. At the rate of 0.75 the block length of the punctured GT-4 code is already 33.3% shorter as compared to the reference code and yet the punctured GT-4 is worse by a relatively low margin of 0.6 dB.

Additionally, we show the performance of the reference code at the rate of 0.75 with the maximum number of iterations raised to 22. As expected from the results in Figure 5 the performance gain is relatively small at 0.2 dB, while raising the maximum number of iterations from 12 to 22 in case of the punctured code resulted in performance gain of almost 1 dB (see Figure 6).

Another big advantage of the punctured GT-4 codes over the reference codes is that they are rate-compatible and thus allow the usage of the type-II hybrid ARQ protocol. This way the overall system throughput can be further increased.

V. CONCLUSION

Overall, the RCP-LDPC codes have proved to be an excellent choice for future communication systems that require rate-adaptability and low complexity. We propose an efficient way to design the mother code and to puncture it such that good performance is maintained over a wide range of rates. In comparison with the unpunctured codes, the RCP-LDPC codes need more iterations to be decoded successfully, but on the other hand they require merely one encoder/decoder pair for operation at all given rates and enable the use of the type-II hybrid ARQ protocol. They yield good performance over a wide range of rates and at the same time reduce the overall system complexity significantly as compared to independently designed unpunctured codes.

REFERENCES

- [1] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCP codes) and their applications," *IEEE Trans. Commun.*, vol. 36, pp. 389–400, Apr. 1988.
- [2] R. G. Gallager, *Low-Density Parity Check Codes*. Cambridge, MA: MIT Press, 1963.
- [3] M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 533–547, Sept. 1981.
- [4] J. Ha, J. Kim, D. Klinc, and S. W. McLaughlin, "Rate-compatible punctured low-density parity-check codes with short block lengths," *submitted to IEEE Trans. Inform. Theory*.
- [5] J. Ha, J. Kim, and S. W. McLaughlin, "Puncturing for finite length low-density parity-check codes," in *Proc. Int. Symp. Information Theory*, Chicago, Illinois, July 2004, p. 151.
- [6] —, "Rate-compatible puncturing of low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. IT-50, pp. 2824–2836, Nov. 2004.
- [7] X. Hu, E. Eleftheriou, and D. M. Arnold, "Progressive edge-growth Tanner graphs," in *Proc. IEEE GLOBECOM*, San Antonio, Texas, Nov. 2001, pp. 995–1001.
- [8] A. Batra, "Multiband ofdm physical layer proposal for IEEE 802.15 task group 3a," Mar. 2004.
- [9] A. F. Molisch, J. R. Foerster, and M. Pendergrass, "Channel models for ultrawideband personal area networks," *IEEE Wireless Communications*, vol. 10, pp. 14–21, Dec. 2003.