DECODING LDPC CODES OVER BINARY CHANNELS WITH ADDITIVE MARKOV NOISE

Chris Nicola, Fady Alajaji and Tamás Linder

Department of Mathematics and Statistics
Queen's University, Kingston, ON K7L 3N6, Canada
{nicola, fady, linder}@mast.queensu.ca

ABSTRACT

We study the design of a belief propagation decoder for low-density parity check (LDPC) codes over binary channels with additive M^{th}-order Markov noise, specifically the queue-based channel (QBC) [1]. The decoder is an extension of the sum product algorithm (SPA) for the case of memoryless channels and is similar to the one developed for the Gilbert-Elliot channel [2, 3, 4]. Simulation results of randomly generated LDPC codes over the QBC show a significant improvement over the standard decoding method for the "equivalent memoryless" channel (resulting from ideal interleaving). We also compare with the decoding method for the GEC for cases where the QBC and GEC are statistically close.

1. INTRODUCTION

There have been several papers published recently on the topic of LDPC codes focusing on performance over channels with memory (e.g., [2, 3, 4]). These papers discuss the decoding of LDPC codes on channels based on hidden Markov models (HMMs), particularly the Gilbert-Elliot Channel (GEC) [5]. These channel models are simple and widely used, but they have an infinite noise memory. Thus, they may not always be good models for many kinds of real world channels where the effect of memory is finite.

The finite-memory Polya contagion channel [6] and the more general queue-based channel (QBC) [1], are based on an M^{th}-order Markov noise processes, where the state of the channel is fully characterized by the last M noise symbols. For these channels, each state transition corresponds to an error or no-error event. With the GEC state transitions may or may not occur regardless of whether or not an error occurs. Both the Polya channel and the QBC have closed-form solutions for the noise steady state and block distributions and for capacity, making them good models for mathematical analysis. For HMM-based channels with memory (such as the GEC) the capacity must be estimated numerically as no closed-form solution exists.

In this work, we present a modified version of the sum-product algorithm for decoding LDPC codes over the QBC using a method similar to the one used in [2] for the GEC. We generalize this design for all binary channels with M^{th}-order Markov noise and known or estimated one-step channel transition probabilities. The decoder is simulated over the QBC and compared with results for the binary symmetric channel (BSC) and the GEC (with parameters chosen to statistically match the QBC). This is done to show the effectiveness of the design as well as the performance gain over a decoder which uses an ideal interleaver to create a memoryless channel.

This work was supported in part by NSERC of Canada and PREA of Ontario.

2. CHANNEL DESCRIPTION

The QBC [1] is a binary finite-memory Markov channel characterized by four channel parameters ε, α, p, and M. The noise process is based on a length-M queue, $\mathbf{q} = (q_1, q_2, ..., q_M)$, which contains the last M channel noise symbols. The state of the channel is characterized by the binary sequence within the queue. The channel output at time i is given by $y_i = x_i \oplus e_i$, where $\oplus$ denotes addition modulo-2, x_i is the input symbol and e_i is the noise symbol. The input and noise processes are assumed to be independent of each other. At time i, e_i is chosen either from the queue with probability ε or from a BSC process with probability $1 - \varepsilon$.

- If e_i is chosen from the queue, then an entry from the queue is chosen randomly such that,

$$p(e_i = q_j) = \begin{cases} 1/(M-1+\alpha) & j = 1...M-1, \\ \alpha/(M-1+\alpha) & j = M. \end{cases}$$

- If e_i is chosen from the BSC process, then,

$$p(e_i = 1) = p.$$

After each transmission the entries of the queue are shifted to the right by one and e_i becomes the new first entry in the queue, q_1. Thus, there are only two possible state transitions from any one state corresponding to $e_i = 0$ and $e_i = 1$.

The channel bit error probability (CBEP) for the QBC is p and the length of the memory is M, the length of the queue. The channel is described in detail in [1] where the authors have also derived closed form expressions for the noise steady-state distribution, the channel block distribution and the channel capacity.

3. DECODER DESIGN

The standard SPA decoder works by passing messages on the Tanner-factor graph of the LDPC code (Fig. 1). The graph connects nodes representing the code-bits ($x_i's$) to nodes representing the parity-check nodes ($c_i's$) those code-bits participate in

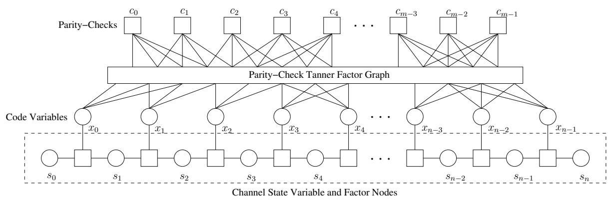

Fig. 1. A generalization of the factor graph for an LDPC code over a channel with memory. The bottom chain in the graph represents the relationship between the channel states.

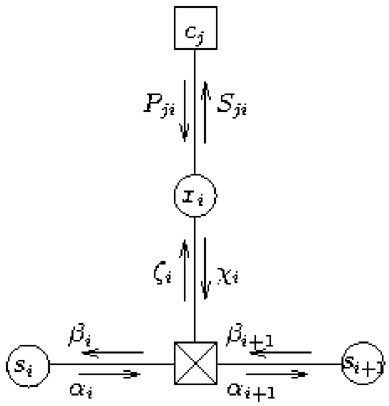

Fig. 2. Diagram showing the six different messages passed by the extended SPA.

(see [7] for a detailed description of the algorithm for memoryless channels). We have modified this algorithm for a channel with memory by factoring the joint distribution for the QBC channel in a similar manner to [2], which considers the GEC. The joint distribution for the QBC is similar to that of the GEC and other finite-state channels and is given by:

$$M_{QBC}(\mathbf{y},\mathbf{x},\mathbf{s}) = P_{\mathbf{Y}}(\mathbf{y}|\mathbf{x},\mathbf{s})P_{\mathbf{S}}(\mathbf{s})P_{\mathbf{X}}(\mathbf{x}), \qquad (1)$$

where $\mathbf{x}$ is a length n codeword vector, $\mathbf{y}$ is the channel output vector of length n and $\mathbf{s}$ is the length $n+1$ state sequence. We note that given $\mathbf{x}$ and $\mathbf{s}$ there can only be one valid choice for $\mathbf{y}$. We can factor equation (1) to obtain the following equation:

$$\begin{aligned} M_{QBC}(\mathbf{y},\mathbf{x},\mathbf{s}) = &\textstyle\prod_{i=0}^{n-1} i(e_i|s_i,s_{i+1}) \\ &p(s_1)\textstyle\prod_{j=0}^{n-1} p(s_{j+1}|s_j)\prod h_k(\mathbf{x}_k). \end{aligned} \qquad (2)$$

We define $h_k(\mathbf{x}_k)$ as the indicator function of the k^{th} parity check and $\mathbf{x}_k$ is the set of variables x_i that participate in the k^{th} parity check. The function $i(e_i|s_i,s_{i+1})$ is an indicator function of whether or not a particular state transition from s_i to s_{i+1} corresponds to an error or no-error ($e_i = 1$ or 0) noting that $e_i = x_i \oplus y_i$. The function $p(s_{j+1}|s_j)$ is simply the one-step channel transition probability for the QBC and $p(s_1)$ is assumed to be distributed according to the noise steady-state distribution for the QBC. The extended factor graph is shown in Fig. 1.

We use the above factorization to derive the equations for the messages passed along the extended factor graph, see Fig. 2. The messages passed between parity checks, code-bits and the channel factors are likelihood ratios (LRs),

$$P_{ij}, S_{ji}, \zeta_i, \chi_i \triangleq \frac{P(x_i=0|y_i)}{P(x_i=1|y_i)}.$$

P_{ij} and S_{ji} are computed according to the standard SPA update rules for the BSC:

$$\begin{aligned} S_{ij} &= \zeta_i \textstyle\prod_{k\neq j} P_{ki}, \\ P_{ij} &= \tfrac{1-\prod_{k\neq j}(1-S_{ki})/(1+S_{ki})}{1+\prod_{k\neq j}(1-S_{ki})/(1+S_{ki})}. \end{aligned}$$

The SPA has four additional messages to handle the computation of the channel factor probabilities. χ_i is the LR passed from the code-bit variables and is computed as simply $\chi_i = \prod_k P_{ki}$. This message is converted to an estimate of the probability of error according to

$$P(\chi_i, y_i) = y_i + (-1)^{y_i}(1+\chi_i)^{-1}.$$

The messages passed along the channel factor graph $\vec{\alpha}_i$ and $\vec{\beta}_i$ are vectors where the j^{th} entry is the estimate that the channel is in state j at time i. ζ_i is the LR passed back to the code-bit nodes. The message update rule for these messages is defined using the factorization in equation (2) as follows:

$$\vec{\alpha}_{i+1}[j] = \sum_{k,e_i} \alpha_i[k] P^i_{e_i}(\chi_i,y_i)\mathbf{P}_{kj} i(e_i|s_i=j,s_{i+1}=k),$$

$$\vec{\beta}_i[j] = \sum_{j,e_i} \beta_{i+1}[k] P^i_{e_i}(\chi_i,y_i)\mathbf{P}_{jk} i(e_i|s_i=k,s_{i+1}=j),$$

$$\zeta_i = \left(\frac{\sum_{j,k}\alpha_i[j]\beta_{i+1}[k]\mathbf{P}_{jk} i(e_i=0|s_i=j,s_{i+1}=k)}{\sum_{j,k}\alpha_i[j]\beta_{i+1}[k]\mathbf{P}_{jk} i(e_i=0|s_i=j,s_{i+1}=k)}\right)^{-y_i},$$

where $P^i_{e_i=1}(\chi_i,y_i) = P^i_e(\chi_i,y_i)$ and $P^i_{e_i=0}(\chi_i,y_i) = 1 - P^i_e(\chi_i,y_i)$ and $\mathbf{P}$ is the one-step channel-state transition matrix. $\vec{\alpha}_i[j]$ and $\vec{\beta}_i[j]$ are the entries of $\vec{\alpha}_i$ and $\vec{\beta}_i$ that correspond to state j.

We now define $\mathbf{E}$ and $\mathbf{C}$ to be the matrices of the one-step channel transition probabilities for the case where there is an error and the case where there is not an error, respectively. Thus $\mathbf{C}+\mathbf{E}=\mathbf{P}$, where $\mathbf{P}$ is the complete one-step channel-state transition matrix. We can rewrite the update rule above using these matrices and the messages passed as:

$$\vec{\alpha}_{i+1} = \frac{(1-P^i_e)\mathbf{C}^T\vec{\alpha}_i + P^i_e\mathbf{E}^T\vec{\alpha}_i}{\vec{u}^T_{2^M}[(1-P^i_e)\mathbf{C}^T\vec{\alpha}_i + P^i_e\mathbf{E}^T\vec{\alpha}_i]},$$

$$\vec{\beta}_i = \frac{(1-P^i_e)\mathbf{C}\vec{\beta}_{i+1} + P^i_e\mathbf{E}\vec{\beta}_{i+1}}{\vec{u}^T_{2M}[(1-P^i_e)\mathbf{C}\vec{\beta}_{i+1} + P^i_e\mathbf{E}\vec{\beta}_{i+1}]},$$

$$\zeta_i = \left(\frac{\vec{\alpha}_i^T\mathbf{C}\vec{\beta}_{i+1}}{\vec{\alpha}_i^T\mathbf{E}\vec{\beta}_{i+1}}\right)^{-y_i}.$$

As we noted above, the QBC has only two types of channel transitions from any state: one if an error occurs and one if no-error occurs. Thus both $\mathbf{C}$ and $\mathbf{E}$ are sparse with only 2^M non-zero entries in each matrix. This significantly reduces the computation needed on the channel sub-graph, making the QBC, and other similar models, a good choice of channel for this type of decoding algorithm.

These update rules can be applied to a more general family of channels with M^{th}-order Markov noise. Any channel that has 2^M states where state transitions correspond to either an error or no error can use this decoder design.

The algorithm proceeds in the following steps:

1. Initialization:

   (a) Initialize the code bit-to-check node messages according the the average probability of error:

   $$S_{ij} = \left(\frac{1-p}{p}\right)^{-y_i}$$

   (b) Initialize the first message of the channel graph $\vec{\alpha}_1$ with the steady-state distribution of the channel.

2. Repeat until codeword estimate passes the parity-check or until N_{max} iterations have been performed:

   (a) Compute all P_{ij}'s for each parity-check node.

   (b) Compute the extrinsic message χ_i for each code-bit and use the threshold ≥ 0 to decide if the bit is a 'zero'.

   (c) Check the current estimate of the codeword against the parity check matrix and stop if it is a valid codeword.

   (d) Compute the forward messages of the channel graph and set the first backward messages to the last forward message (i.e., $\beta_{n+1} = \alpha_{n+1}$).

   (e) Compute the backward messages of the channel graph and set the first forward message to the last backward message (i.e., $\alpha_1 = \beta_1$).

   (f) Compute the ζ_i messages for each code-bit.

   (g) Compute all the S_{ij}'s for each code bit.

The SPA employs the same belief propagation method used to decode Turbo codes, but because LDPC codes are defined by their parity-check matrix, we can use the parity-check to determine, after each iteration, if our estimate is valid (unlike the case of Turbo codes). Additionally, since the parity-check matrix is sparse by definition, the parity-check validation can be performed with reduced complexity.

4. RESULTS

The LDPC codes used in the simulations were generated randomly and all are rate 1/2 with 3 checks per bit and 6 bits per parity-check (regular-LDPC codes). All cycles of minimum length (cycles of length 4) in their Tanner graph are removed to improve convergence of the decoding algorithm. Short cycles can affect the performance of the algorithm by making messages in separate iterations highly dependent [7].

In Fig. 3, we show the performance of the LDPC codes over the QBC compared with its ideally interleaved (memoryless) version (BSC). For these simulations we have fixed $\alpha = 1$, $M = 4$ and ε is chosen for each p so as to ensure a fixed channel noise correlation value of 0.5. We note a significant improvement due to exploiting the channel memory at the decoder when compared

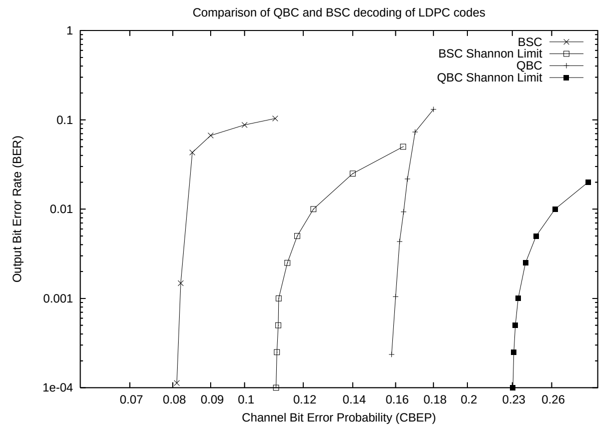

Fig. 3. Plot of SPA decoding the BSC and QBC and their respective Shannon limits. The code length is 100,000 and the maximum iterations is 200.

to using an interleaver. But there remains a considerable performance gap to the QBC Shannon limit. The QBC achieves a BER of 10^{-4} at a CBEP $\cong 0.157$ where the Shannon limit for this QBC at a BER of 10^{-4} is 0.2304. For the BSC, we achieve a BER 10^{-4} at CBEP $\cong 0.080$ and the Shannon limit is 0.11. From Fig. 3, we however remark that, proportionally, there is little difference in the performance gap for the two channels. The QBC decoder achieved 68% of the Shannon limit while the BSC decoder achieved 72%.

In [8], the authors show that for a considerable range of channel conditions, the QBC can be designed to be nearly statistically identical to the GEC. In Table 1, results are shown comparing the QBC to the GEC for two sets of parameters taken from [8]. The performance of the two decoders is very close. Additionally, these results demonstrate that an improvement in performance over the memoryless channel can be achieved when using a channel model to model the noise process of another channel, which is of interest for the real-world communication environment, as there is always a mismatch between the real channel and its adapted model.

5. CONCLUSIONS

We have demonstrated how the sum product algorithm can be modified to decode binary channels with additive M^{th}-order Markov noise. The benefit of considering the channel memory is demonstrated experimentally, showing a significant gain over the memoryless strategy of employing an interleaver. In fact, we observe that for the same code, one can expect close to the same performance relative to capacity for the QBC as is obtained for the BSC.

The extended SPA decoder for Markov channels has the advantage of operating only on the extrinsic information (χ_i) of the standard SPA decoder. Thus, it could be operated in parallel to the standard decoder using separate hardware. A channel factor computer could read the extrinsic information being stored by the decoder, compute the forward-backward calculations, and then write new channel error probabilities for the standard de-

Channels Parameters

GEC	CBEP	P_g	P_b	g	b
Exp. 1	0.09	0.0519	0.6118	0.0450	0.0033
Exp. 2	0.08	0.0439	0.5746	0.0450	0.0033

QBC	CBEP	ε	α	p	M
Exp. 1	0.09	0.5705	0.4168	0.0900	5
Exp. 2	0.08	0.5711	0.4312	0.0800	5

Results

BER	GEC	QBC	GEC w/ QBC Decoder	QBC w/ GEC Decoder	BSC
Exp. 1	1.6E-05	<10^{-5}	1.9E-03	5.8E-04	6.5E-02
Exp. 2	<10^{-5}	<10^{-5}	1.0E-05	1.7E-05	6.5E-03

Table 1. Results from comparison of decoding on the GEC (with parameters P_g, P_b, g and b), the QBC channel that approximates the GEC, the GEC using the QBC decoder (and vice-versa) and the BSC. For this simulation a length 10,000 code was used with a maximum of 200 decoding iterations.

coder to use on its next iteration. Such a design would allow for inexpensive modification of existing systems and hardware already designed to use the SPA for decoding.

Through simulations of the QBC and GEC channel models, we note that a channel model can be matched to the statistics of another channel to offer a decoding gain over the memoryless strategy. This suggests that simple models for channels with memory can be used to design improved decoders for real-world channels though statistical analysis and channel matching, or through decoders which perform channel parameter estimation based on the received data. The results indicate that while such a strategy is not perfect, it could offer a notable improvement over existing memoryless strategies.

Despite strong demonstrated performance, there is still a significant performance gap vis-a-is the Shannon limit that is left unexploited. In [9], the authors use the technique of density evolution [10] to design irregular LDPC codes for the AWGN channel with near capacity-achieving performance. This technique is extended in [2, 11] to the GEC and the Polya-contagion channel for regular LDPC codes and then in [12] for irregular LDPC codes over the GEC. Design and analysis of irregular LDPC codes for the QBC using density evolution would likely offer significant performance gains allowing the memory of the channel to be exploited further.

Density evolution is unfortunately a complex process even for memoryless channels. However it may be sufficient to perform this analysis on simple channel models. An irregular code that demonstrates significant performance gains on a BSC channel using the standard SPA decoder may also offer comparable gains on a channel with memory, such as the QBC, if it is decoded appropriately to exploit the channel memory. Exploring the performance of irregular LDPC codes over the QBC will be the subject of future work.

6. REFERENCES

[1] L. Zhong, F. Alajaji, and G. Takahara, "A queue-based model for binary communication channels," in *Proc. Forty-First Annual Allerton Conference on Communication, Control and Computing*, Monticello, IL, October 2003.

[2] A. W. Eckford, F. R. Kschischang, and S. Pasupathy, "Analysis of LDPC codes in channels with memory," in *Proc. 21st Queen's Biennial Symposium on Communications*, Kingston, Ontario, Canada, June 2002.

[3] E. Ratzer, "Low-density parity-check codes on Markov channels," in *Proc. Second IMA Conference on Mathematics in Communications*, December 2002.

[4] J. Garcia-Frias, "Decoding of low-density parity-check codes over finite-state binary Markov channels," *IEEE Trans. Commun.*, vol. 52, no. 11, pp. 1840–1843, November 2004.

[5] M. Mushkin and I. Bar-David, "Capacity and coding for the Gilbert-Elliott channel," *IEEE Trans. Inform. Theory*, vol. 35, no. 6, pp. 1277–1290, November 1989.

[6] F. Alajaji and T. Fuja, "A communications channel modeled on contagion," *IEEE Trans. Inform. Theory*, vol. 40, no. 6, pp. 2035–2041, November 1994.

[7] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, February 2001.

[8] L. Zhong, F. Alajaji, and G. Takahara, "An approximation of the Gilbert-Elliott channel via a queue-based channel model," in *Proc. IEEE International Symposium on Information Theory*, Chicago, June-July 2004.

[9] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 619–637, February 2001.

[10] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 599–618, February 2001.

[11] V. Nagarajan and O. Milenkovic, "Performance analysis of structured LDPC over the Polya-urn channel with finite memory," in *Proc. Canadian Conf. Electrical Computer Engineering*, Niagara Falls, Canada, May 2004.

[12] A. W. Eckford, F. R. Kschischang, and S. Pasupathy, "Designing very good low-density parity-check codes for the Gilber-Elliot channel," in *Proc. 8th Canadian Workshop on Information Theory*, Waterloo, Ontario, Canada, May 2003.