A Comparison of Decoding Latency for Block and Convolutional Codes[*]

Moritz Kaiser
Institute for Communications Engineering
Technische Universität München
80333 Munich, Germany
Email: moritz.kaiser@tum.de

Wai Fong
Goddard Space Flight Center/NASA
Code 567
Greenbelt, MD, USA 20771
Email: wai.fong@gsfc.nasa.gov

Marcin Sikora
and Daniel J. Costello, Jr.
Department of Electrical Engineering
University of Notre Dame
Notre Dame, IN, USA 46556
Email: {msikora|costello.2}@nd.edu

***Abstract*—In this paper we compare the decoding latency, i.e., the delay between the time a channel symbol is received and the time it is decoded, of block and convolutional codes. In particular, we compare low density parity check (LDPC) block codes with iterative message-passing decoding to convolutional codes with Viterbi decoding and stack sequential decoding. On the basis of simulations, we show that, for a code rate of $1/2$, a target bit error rate of 10^{-4}, and an allowed latency of up to approximately 2000 information bits, convolutional codes with stack sequential decoding require a smaller signal-to-noise ratio (SNR) than LDPC codes with iterative message-passing decoding. For larger allowed latencies, the advantage switches to LDPC codes.**

I. Introduction

In 1948 Shannon founded information theory with his article "A mathematical theory of communication" [1], in which he proved that, for a given communication channel, coded transmission with arbitrarily small probability of error is possible at rates below capacity, given long enough codes. Since then communication engineers have tried to develop error-correcting codes that achieve a small probability of error at rates as close to channel capacity as possible. In the process, many important codes were discovered, such as Hamming codes [2], Golay codes [3], BCH codes [4], [5], Reed-Solomon codes [6], convolutional codes [7], and turbo codes [8]. Then, in 1995, the capacity-approaching class of low density parity check (LDPC) codes, originally introduced by Gallager in [9], was rediscovered by MacKay and Neal [10] and Wiberg et al. [11]. Currently LDPC codes are employed in satellite-based digital video broadcasting and long-haul optical communication standards and are likely to be adopted in the IEEE WLAN standard and third-generation mobile telephony.

In practical communication systems, a low error probability and a high transmission rate are not the only important factors. The complexity and memory requirements of the encoder and decoder influence the cost of a device, such as a mobile phone. Another very important parameter is the latency, i.e., the time it takes to recover the transmitted message. This delay is introduced by the encoder, the decoder, and the channel and has always been crucial for telephony, since high latency can seriously handicap a voice conversation. Also more recent applications like video conferencing and remote control have demanding latency requirements.

Communication engineers largely agree that for applications not requiring low latencies, long LDPC codes are the right method to achieve capacity-approaching performance [12]. But there is currently no consensus regarding the right coding method to use for low required latencies. In this paper, we compare the performance of convolutional codes to block codes on the basis of an equal latency constraint, with particular emphasis on the low latency case.

The paper is organized as follows. In Section II we define decoding latency, and in Section III we introduce decoding speed, a parameter needed to compute latency. The results of simulations are presented in Section IV and we directly compare LDPC block codes to convolutional codes in Section V. Section VI concludes the paper.

II. Decoding Latency

We consider a simplified transmission system as depicted in Fig. 1. The overall latency is defined as the difference between the time the source emits an information bit and the time the information bit is decoded.

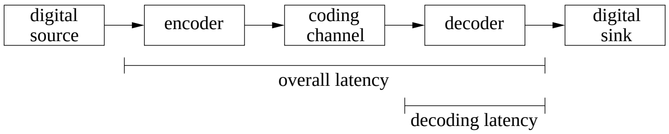


Fig. 1. The definition of latency.

The decoding latency is denoted by l and is defined as the overall latency minus the encoding time and the channel delay. We measure time in terms of information bits. One information bit corresponds to the time the source needs to emit one bit, and thus time can easily be converted from information bits to seconds.

We now define decoding latency more precisely for the three decoding methods we want to compare, namely LDPC

[*] This work was partially supported by NASA grant NNX 07AK53G.

From: Proc. ISCTA '09, Ambleside, UK

block codes with iterative message-passing decoding [13] and convolutional codes with Viterbi decoding [14] and stack sequential decoding [15].

LDPC block codes with iterative message-passing decoding: In general, the decoder must wait for the whole block (K information bits for an (N, K) LDPC block code) to arrive before it can start decoding. Blocks are then decoded by an iterative message-passing decoder that employs a stopping rule and a buffer (see, e.g., [16]). The decoding itself, along with the possible buffering of some blocks, requires additional time, referred to as the computational time $t^{\text{ldpc}}_{\text{comp}}$. The number of decoding iterations per block and the time a block waits in the buffer vary with the channel quality. Thus $t^{\text{ldpc}}_{\text{comp}}$ is a random variable and we consider the average computational time $\bar{t}^{\text{ldpc}}_{\text{comp}}$.

The average decoding latency $\bar{l}^{\text{lpdc}}$ of LDPC block codes equals the arrival time t_{block} of one incoming block plus the average computational time $\bar{t}^{\text{ldpc}}_{\text{comp}}$ needed for decoding and possible buffering of the block, i.e.,

$$\bar{l}^{\text{ldpc}} = t_{\text{block}} + \bar{t}^{\text{ldpc}}_{\text{comp}}.$$

If $\bar{t}^{\text{ldpc}}_{\text{comp}}$ is less than t_{block}, one decoder is sufficient. If $\bar{t}^{\text{lpdc}}_{\text{comp}}$ is greater than t_{block}, several decoders must be applied in parallel, as shown in Fig. 2; otherwise, the buffer for incoming blocks will fill up. The number D of required decoders can be computed as $D = \left\lceil \frac{\bar{t}^{\text{lpdc}}_{\text{comp}}}{t_{\text{block}}} \right\rceil$, where $\lceil \cdot \rceil$ is the ceiling function.

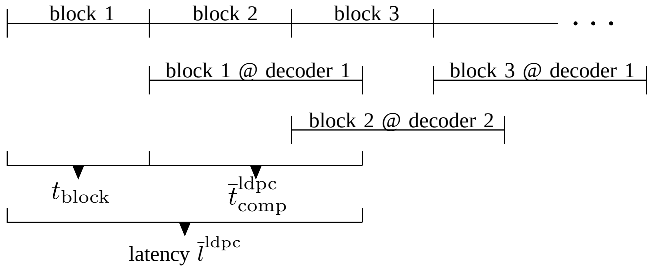


Fig. 2. Decoding latency of LDPC block codes for $\bar{t}^{\text{ldpc}}_{\text{comp}} = 1.5 \cdot t_{\text{block}}$.

Convolutional codes with Viterbi decoding: In contrast to iterative message-passing decoding of LDPC block codes, a convolutional decoder can begin its computations after t_{sect}, the arrival time of one incoming trellis (or tree) section (k information bits for an (n, k, m) convolutional code). After having received a trellis section, a Viterbi decoder computes the new state metrics in the trellis. The number of computations per trellis section is fixed (2^{km}), and thus the computational time $t^{\text{vit}}_{\text{comp}}$ required to compute the new state metrics is constant.

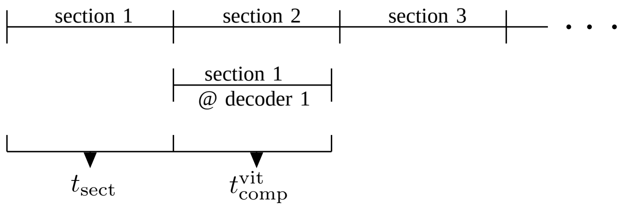


Fig. 3. Viterbi decoding for $t^{\text{vit}}_{\text{comp}} = 0.9 \cdot t_{\text{sect}}$.

Parallel decoders analogous to iterative message-passing decoding of LDPC block codes cannot be employed because the metrics of one trellis section must be known in order to compute the metrics of the next section[1]. The decoder must finish before the code bits of the next trellis section arrive, i.e., $t^{\text{vit}}_{\text{comp}}$ cannot exceed t_{sect} (as shown in Fig. 3); otherwise, incoming code bits will be lost.

Viterbi decoders cannot store paths of infinite length. Thus a finite path memory τ is employed, i.e., after $k\tau$ information bits a decision is forced. The decoding latency is then the time required to receive τ trellis sections ($k\tau$ information bits $\doteq \tau \cdot t_{\text{sect}}$) plus the computational time $t^{\text{vit}}_{\text{comp}}$ for one trellis section, as depicted in Fig. 4, i.e.,

$$l^{\text{vit}} = \tau \cdot t_{\text{sect}} + t^{\text{vit}}_{\text{comp}}.$$

Finally, we note that there is a tradeoff between τ, which directly influences the latency, and the decoded bit error rate (BER) (see, e.g., [17]).

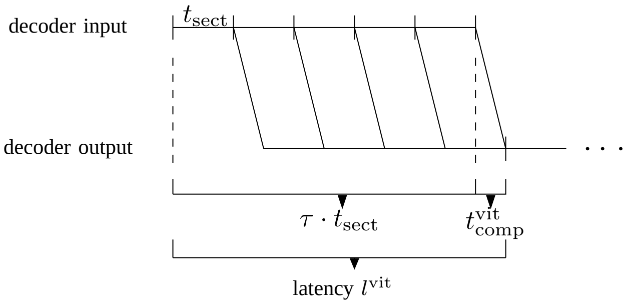


Fig. 4. Decoding latency for a Viterbi decoder.

Convolutional codes with stack sequential decoding: After having received a tree section, a stack sequential decoder extends the most promising path, i.e., the top path in the stack. When the top path includes the most recently received tree section, it waits until the next tree section has been received before proceeding. In general, the number of path extensions per incoming tree section varies with the channel quality, and thus the computational time $t^{\text{stack}}_{\text{comp}}$ per tree section is a random variable, and buffering is required (see, e.g., [15]). Hence we consider the average computational time $\bar{t}^{\text{stack}}_{\text{comp}}$.

If the decoder operates close to the computational cutoff rate, the average number of path extensions per tree section, and thus $\bar{t}^{\text{stack}}_{\text{comp}}$, can become greater than t_{sect}, causing the

[1] Parallel decoders can be employed, however, to reduce $t^{\text{vit}}_{\text{comp}}$. But the decoders still only work on one trellis section at a time, and hence we consider this as a single fast decoder.

buffer to overflow and incoming code bits to be lost. This can be prevented by discarding corrupted tree sections after a certain number of computations (see, e.g., [18]).

After $k\tau$ information bits a decision is forced, because the buffer cannot store paths of infinite length. For stack sequential decoding, τ is called the backsearch limit. The average decoding latency of stack sequential decoding is then

$$\overline{l}^{\text{stack}} = \tau \cdot t_{\text{sect}} + \overline{t}^{\text{stack}}_{\text{comp}}.$$

As with Viterbi decoding, the BER decreases but the latency increases with increasing τ.

III. Decoding Speed

In order to be able to determine the computational time for the three decoding techniques, we define the decoding speed s as the average number of information bits the decoder can decode per incoming information bit

$$s = \frac{\text{number of decoded inf. bits}}{\text{incoming inf. bit}}.$$

In Section II, we have seen that the decoding speed s^{vit} of Viterbi decoders and the average decoding speed $\overline{s}^{\text{stack}}$ of stack sequential decoders must be at least one, i.e., $s^{\text{vit}}, \overline{s}^{\text{stack}} \in [1, \infty)$, if we do not want to lose incoming code bits. (Note that, for stack sequential decoding, we must consider the average decoding speed, since s^{stack} is a random variable.) In our analysis, we assume the slowest possible decoding speed for the decoding of convolutional codes ($s^{\text{vit}} = 1$ and $\overline{s}^{\text{stack}} = 1$), since it is not likely that faster hardware than needed is used in a decoder.

We have also seen in Section II that the average decoding speed $\overline{s}^{\text{ldpc}}$ of an iterative message-passing decoder for LDPC block codes can be less than one, provided that we have enough decoders. For a completely fair comparison, we would have to determine the average block decoding speed $\overline{s}^{\text{ldpc}}$ assuming the same hardware resources employed for convolutional codes. But computing $\overline{s}^{\text{ldpc}}$ under these conditions is not feasible, since many factors, such as code rate, channel quality, and implementation architecture, would have to be considered. For this reason we treat $\overline{s}^{\text{ldpc}}$ as a variable.

IV. Results

All simulations were performed using rate $1/2$ codes on an additive white Gaussian noise (AWGN) channel and we assumed binary phase-shift-keyed (BPSK) modulation. (Choosing other code rates does not fundamentally change the reported comparisons.)

LDPC block codes with iterative message-passing decoding: We implemented an LDPC iterative message-passing decoding algorithm that has a maximum number of iterations equal to 50. The parity check matrices are taken from the rate $1/2$ LDPC codes listed in Appendix A of MacKay's Encyclopedia of Sparse Graph Codes [19]. (There are many more up-to-date sources of good LDPC codes. However, for the short block lenghts considered, the choice of code has only a minor effect on the results.) We measured the BER for various block lengths and normalized signal-to-noise ratios (SNRs) E_b/N_0, which gave us a set of BER vs. SNR curves for different block lengths. Subsequently, we interpolated these curves at a target BER $= 10^{-4}$ and drew the required SNR E_b/N_0 as a function of the block length K, as shown in Fig. 5. (Different target BERs can also be considered, but they do not substantially alter the conclusions.)

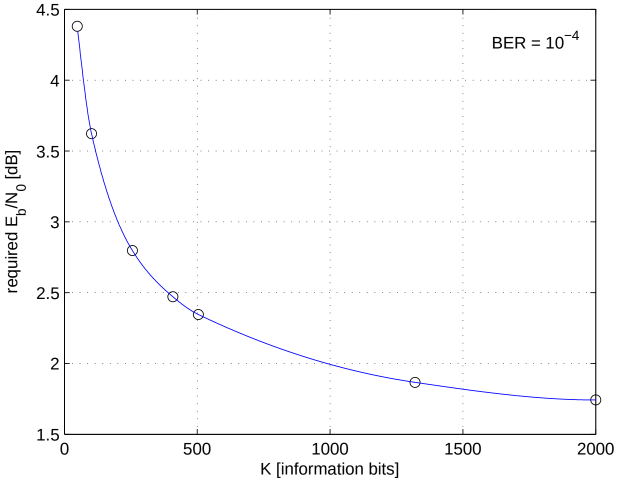


Fig. 5. Iterative message-passing decoding: Required SNR E_b/N_0 as a function of block length K for a target BER $= 10^{-4}$.

Convolutional codes with Viterbi decoding: For Viterbi decoding, we generated BER vs. SNR curves for several path memories τ and code memories m. Optimal rate $1/2$ codes were taken from [17]. We accounted for the influence of a finite path memory τ by deciding whether a certain information bit is a '0' or a '1' after τ further information bits were received. We again interpolated these curves at a target BER $= 10^{-4}$ and plotted the required SNR as a function of τ in Fig. 6. Note that, in practice, codes with memory m greater than 12 are not feasible, since the number of state metrics (2^m) that must be computed at every time step grows exponentially with m.

Convolutional codes with stack sequential decoding: We performed the same simulations for the stack algorithm as for Viterbi decoding and the results are shown in Fig. 7. Note that the curves do not improve any further for code memories m greater than or equal to 16. It is explained in [15] that, if a stack sequential decoder operates at rates below the computational cutoff rate, the average number of path extensions per arriving information bit can be upper bounded, whereas if it operates above the cutoff rate this number can become prohibitively large. Thus, since we limited the maximum number of path extensions per incoming information bit to 250, the required SNR cannot be made arbitrarily small by increasing m.

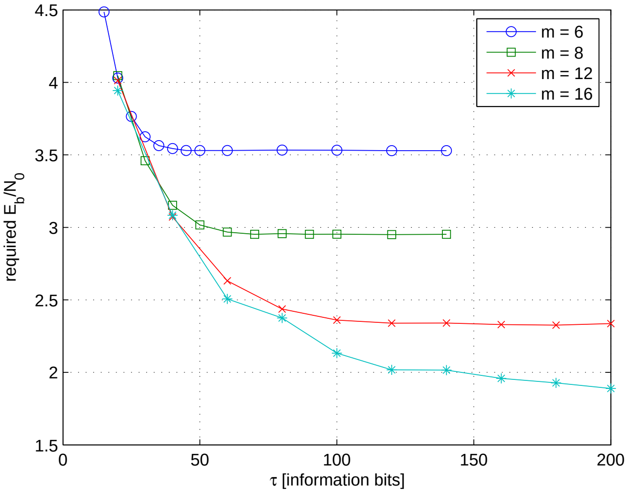


Fig. 6. Viterbi decoding: Required SNR E_b/N_0 as a function of path memory τ for a target BER $= 10^{-4}$.

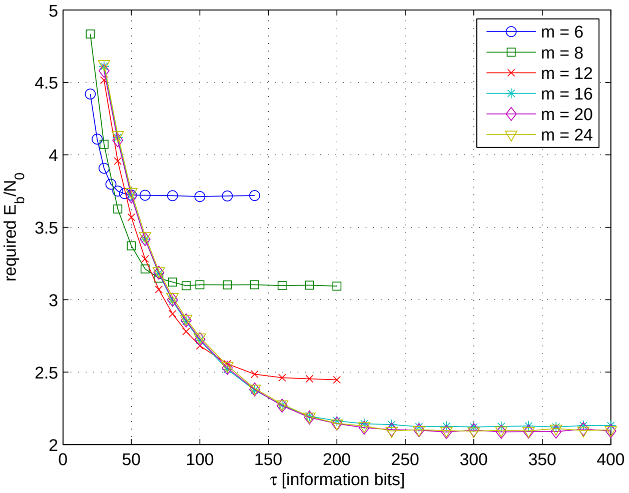


Fig. 7. Stack sequential decoding: Required SNR E_b/N_0 as a function of backsearch limit τ for a target BER $= 10^{-4}$.

V. Comparison

Now we compare the results for block decoding (iterative message-passing decoding) directly to convolutional decoding (Viterbi or stack sequential decoding), i.e., we combine the curves from Section IV into one graph. Thus we transform both the SNR vs. K and the SNR vs. τ curves into SNR vs. l curves, with l being the decoding latency. Considering the definitions of Sections II and III, the average decoding latency is $\bar{l}^{\mathrm{ldpc}} = K(1+1/\bar{s}^{\mathrm{ldpc}})$ for iterative message-passing decoding and $\bar{l} = k(\tau+1)$ for Viterbi decoding and stack sequential decoding.

As a result, the latency requirements of block and convolutional decoding can be depicted in one figure, and, by considering $\bar{s}^{\mathrm{ldpc}}$ as a variable, we can generate a set of curves for block decoding of LDPC codes, as demonstrated in Figs. 8 and 9. For a certain average decoding speed $\bar{s}^{\mathrm{ldpc}}$, the curves for iterative message-passing decoding and a convolutional decoding method (Viterbi decoding in Fig. 8 and stack sequential decoding in Fig. 9) will intersect at a certain latency l_{int}.

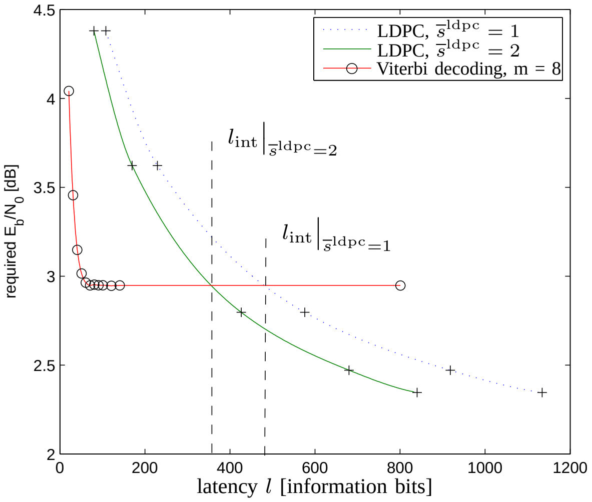


Fig. 8. SNR required to achieve a target BER $= 10^{-4}$ as a function of the latency for Viterbi decoding of convolutional codes and iterative message-passing decoding of LDPC block codes with average decoding speeds $\bar{s}^{\mathrm{ldpc}} = 1$ and $\bar{s}^{\mathrm{ldpc}} = 2$.

In Fig. 10, we depict the l_{int} vs. $\bar{s}^{\mathrm{ldpc}}$ curves for the two convolutional decoding methods. These curves can be interpreted as follows. If we know which average decoding speed $\bar{s}_0^{\mathrm{ldpc}}$ of LDPC block codes corresponds to $s^{\mathrm{vit}} = 1$ (or $\bar{s}^{\mathrm{stack}}$) for our hardware resources and we decide which latency l_0 our application requires, LDPC block codes require less SNR (to achieve the target BER) if the point $(\bar{s}_0^{\mathrm{ldpc}}, l_0)$ lies above the curve and convolutional codes require less SNR if the point lies below the curve. Among the convolutional decoding methods, we see that stack sequential decoding is capable of outperforming Viterbi decoding, since higher code memories are possible.

As mentioned in Section III, it is difficult to determine the exact average block decoding speed $\bar{s}^{\mathrm{ldpc}}$ that corresponds to $s^{\mathrm{vit}} = \bar{s}^{\mathrm{stack}} = 1$. Nevertheless, it is reasonable to assume that $\bar{s}^{\mathrm{ldpc}}$ may be less than $\bar{s}^{\mathrm{stack}}$, since a stack decoder has, if the rate is not greater than the cutoff rate, a relatively low computational effort compared to an iterative message-passing decoder. We consider two cases as examples: (1) If $\bar{s}^{\mathrm{ldpc}} = 0.5$

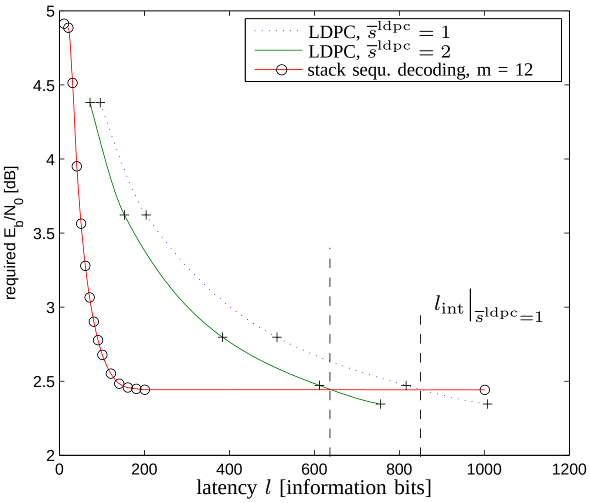


Fig. 9. SNR required to achieve a target BER $= 10^{-4}$ as a function of the latency for stack sequential decoding of convolutional codes and iterative message-passing decoding of LDPC block codes with average decoding speeds $\overline{s}^{\mathrm{ldpc}} = 1$ and $\overline{s}^{\mathrm{ldpc}} = 2$.

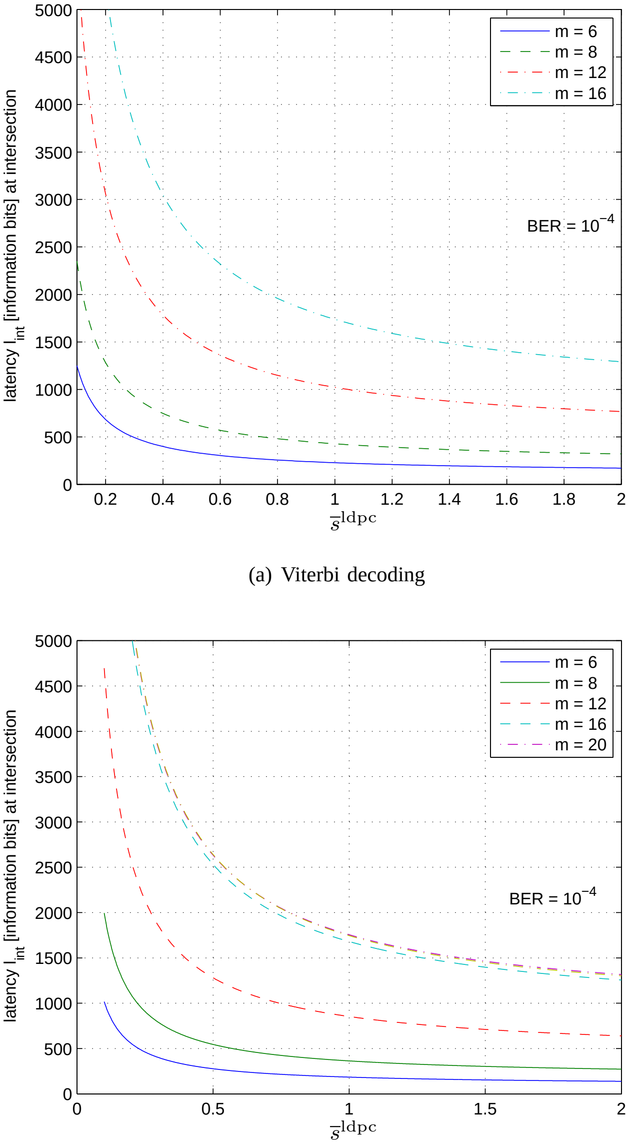
and $\overline{s}^{\mathrm{stack}} = 1$ and we have an application that requires a decoding latency less than about 2500 information bits, we should use convolutional codes with sequential decoding in order to minimize the required SNR necessary to achieve the target BER $= 10^{-4}$; and (2) if $\overline{s}^{\mathrm{ldpc}} = 1.5$ and $\overline{s}^{\mathrm{stack}} = 1$, stack sequential decoding outperforms iterative message-passing decoding only up to a required latency of about 1500 information bits.

As can be seen from Fig. 10, the l_{int} vs. $\overline{s}^{\mathrm{ldpc}}$ curves level off for average block decoding speeds $\overline{s}^{\mathrm{ldpc}} \geq 2$. So, even if we assume that $\overline{s}^{\mathrm{ldpc}}$ is much faster than s^{vit} or $\overline{s}^{\mathrm{stack}}$, there remains a range of required latencies where convolutional decoding requires less SNR than block decoding.

VI. Conclusions

We have shown that, on an AWGN channel with BPSK modulation, and for a code rate of 1/2 and a low fixed allowable latency, stack sequential decoding and Viterbi decoding require a lower SNR to achieve a target BER of 10^{-4} than iterative message-passing decoding of LDPC codes, and that sequential decoding can outperform Viterbi decoding because higher code memories can be employed.

In particular, if we assume $\overline{s}^{\mathrm{stack}} = 1$ and $\overline{s}^{\mathrm{ldpc}} = 0.5$, stack sequential decoding of convolutional codes requires a lower SNR than iterative message-passing decoding of LDPC block codes up to a required decoding latency of about 2500 information bits, and, for $\overline{s}^{\mathrm{stack}} = 1$ and $\overline{s}^{\mathrm{ldpc}} = 1.5$, up to about 1500 information bits. We expect that for rates higher than $1/2$ the comparison between iterative message-passing decoding of LDPC codes and Viterbi or sequential decoding of convolutional codes will remain roughly the same.



(a) Viterbi decoding

(b) Stack sequential decoding

Fig. 10. Latency l_{int} at which convolutional decoding and message-passing decoding require equal SNRs (to achieve a target BER $= 10^{-4}$) as a function of the average block decoding speed $\overline{s}^{\mathrm{ldpc}}$.

Note that if we chose a required BER of less than 10^{-4}, the l_{int} vs. s_{block} curves would move even higher, since the BER vs. SNR curves of large memory convolutional codes are generally steeper than those of moderate length LDPC codes.

We also found that the new look-ahead sequential decoding algorithm, introduced in [20], can outperform stack sequential decoding, but results have been obtained only for a binary symmetric channel (BSC). For details, see [21].

References

[1] C. E. Shannon, “A mathematical theory of communication,” *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, July 1948.

[2] R. W. Hamming, “Error detecting and error correcting codes,” *Bell Systems Technology J.*, vol. 29, pp. 147–160, April 1950.

[3] M. J. E. Golay, “Notes on digital coding,” *Proc. I. R. E.*, vol. 37, p. 657, 1949.

[4] R. C. Bose and D. K. Ray-Chaudhuri, “On a class of error-correcting binary group codes,” *Infom. and Contr.*, vol. 3, pp. 68–79, March 1960.

[5] A. Hocquenghem, “Codes correcteurs d’erreurs,” *Chiffres*, vol. 2, pp. 147–156, September 1959.

[6] I. S. Reed and G. Solomon, “Polynomial codes over certain finite fields,” *J. Soc. Ind. Appl. Math.*, vol. 8, pp. 300–304, June 1960.

[7] P. Elias, “Coding for noisy channels,” *IRE Conv. Rec.*, vol. 4, pp. 37–49, 1955.

[8] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near shannon limit error-correcting coding and decoding: Turbo codes,” *IEEE Trans Commun.*, vol. COM-44, pp. 1261–1271, October 1996.

[9] R. G. Gallager, “Low density parity-check codes,” *IRE Transactions Information Theory*, vol. IT-8, pp. 21–28, January 1962.

[10] D. J. C. MacKay and R. M. Neal, “Near shannon limit performance of low density parity check codes,” *IEE Electronic Letters*, vol. 32, no. 18, pp. 1645–1646, August 1996.

[11] N. Wiberg, N.-A. Loeliger, and R. Kötter, “Codes and iterative decoding on general graphs,” *Eur. Trans. Telecommun.*, vol. 6, pp. 513–526, June 1995.

[12] K. Andrew, D. Divsalar, S. Dolinar, and J. Hamkins, “The development of turbo and LDPC codes for deep space applications,” *Proc. IEEE*, vol. 95, no. 11, November 2007.

[13] F. Kschischang, B. Frey, and H.-A. Loeliger, “Factor graphs and the sum-product algorithm,” *IEEE Transactions on Information Theory*, vol. IT-47, no. 2, pp. 498–519, Feb 2001.

[14] A. J. Viterbi, “Error bounds for convolutional codes and an asumprotically optimum decoding algorithm,” *IEEE Transactions Information Theory*, vol. IT-13, pp. 260–269, April 1967.

[15] R. Johannesson and K. S. Zigangirov, *Fundamentals of Convolutional Coding*. Wiley-IEEE Press, 1999.

[16] S. L. Fogal, S. Dolinar, and K. Andrews, “Buffering requirements for variable-iterations ldpc decoders,” Submitted to ISIT, 2007.

[17] S. Lin and D. J. Costello, Jr., *Error Control Coding*, 2nd ed. Prentice Hall, 2004.

[18] F. Wang and D. J. Costello, Jr., “Erasure-free sequential decoding of trellis codes,” *IEEE Trans. Inform. Theory*, vol. IT-40, pp. 1803–1817, November 1994.

[19] D. J. C. MacKay, “Encyclopedia of sparse graph codes,” http://www.inference.phy.cam.ac.uk/mackay/codes/data.html, cavendish Laboratory, University of Cambridge, UK.

[20] M. Sikora and D. J. Costello, Jr., “Sequential decoding with look-ahead path metric,” in *Proc. IEEE Inter. Symp. Inform. Theory (ISIT’07)*, Seattle, WA, July 2007.

[21] M. Kaiser, “A comparison of decoding latency for block and convolutional codes,” Master’s thesis, Technische Universität München, 80333 Munich, Germany, April 2008.