

Marker codes for channels with insertions and deletions

Edward A. Ratzer

Inference Group, Cavendish Laboratory,
University of Cambridge, Cambridge CB3 0HE, UK
ear23@mrao.cam.ac.uk

Abstract

Coding for channels with synchronization errors is studied. Marker codes, each consisting of a low-density parity-check code with inserted markers, are developed. At low insertion-deletion probabilities marker codes are shown to outperform watermark codes. Full iterative decoding enhances performance to close to the capacity bounds. The low-density parity-check codes are optimized and the best known rate $R = 0.5$ code for the insertion-deletion channel presented. The codes are also shown to be effective on the bit-deletion channel.

1 Introduction

In this paper we look at coding for channels with synchronization errors. Such channels are common, examples include:

Serial line The clock speed of the transmitter may not be accurately known (for instance due to temperature variations in the clock) so the time of arrival of each transmitted bit is not known.

Hard disc Variations in the rotation speed (for instance due to mechanical vibrations or shock) mean the position of the head relative to the platter may be uncertain.

DAT tape Tape stretch leads to problems similar to those suffered by a hard disc.

We follow an information theory convention and model such a channel by random uncorrelated insertion or deletion events at unknown positions. A flowchart of the channel model is shown in figure 1. The capacity for channels of this kind is not known exactly. A capacity lower bound from [26] and an upper bound from [24] are used in this paper.

Marker codes [11] were originally designed to be able to deal with single insertion or deletion error events. The bit stream to be transmitted has a regular marker (or header) inserted in it. For example the marker ‘001’ may be inserted between every 4 data bits:

$$01101100101010 \mapsto 01100011100001101000110$$

The decoder can look for the markers and use any shift in their position to deduce bit loss or gain. Errors in the matched sequence can then be corrected with a conventional code. With advances in computer power probabilistic sequence matching, as described below, can be carried out. The coding system is shown in figure 2. We use a low-density parity-check outer code; such a code is ideally suited to incorporation in a probabilistic message-passing scheme and does not contain local structure.

Watermark codes [6] are a similar scheme, but rather than having bursts of synchronization information and bursts of data, the information is distributed uniformly. To encode, the data bits are uniformly sparsified and then added to a watermark sequence. To decode, probabilistic resynchronization can be carried out with the watermark sequence.

In the literature, watermark codes appear to present the best simulation results. Other coding schemes have been presented which may be of interest for computational or theoretical reasons. A few are listed below:

Comma-free codes The codewords are constructed so that they have the property that no overlap between the codewords can be confused as a codeword [21]. If a codeword is corrupted with an insertion or deletion it is possible to regain synchronization after the error, however error correction within the corrupted codeword is not generally possible [1].

Convolutional coding In [7] a standard convolutional code encoder with a set of normal decoders is used. This leads to a computationally efficient scheme, but the results are disappointing. A modified convolutional encoder and decoder are used in [22] to achieve promising results.

Levenshtein codes In a similar manner to the Hamming distance, the Levenshtein distance [13] between words is the minimum number of insertions or deletions necessary to get from one word to another. Codes exist which try to optimize the minimum Levenshtein distance between codewords but practical codes with good distance properties have not yet been developed.

Schulman and Zuckerman codes Concatenated codes that are asymptotically good are presented in [18] but experimental results are not developed. The codes appear to mainly be of theoretical interest [6].

See [20] for good coverage of older schemes.

Earlier work on marker codes [17] presented an experimental way to optimize the markers based on the capacity of the effective channel illustrated in figure 2. Algebraic optimization of markers has also been attempted [9, 12]. In this paper the results from [17] are extended by comparing complete marker-code-based systems with watermark codes. The benefit of iterative

probabilistic resynchronization is also studied. We show simulation results that outperform the best known results.

2 Probabilistic Resynchronization

We use the forward-backward algorithm to find the conditional probability distribution of the transmitted bits given the received bits. We follow the presentation of [17], modified to include iterative probabilistic resynchronization.

Figure 3(a) shows a representation of synchronization errors. The solid line indicates one particular sequence of insertions and deletions which then defines a mapping from the transmitted bits to the received bits. If there were no insertions or deletions the mapping would follow the dashed line – this represents a one-to-one mapping between the received and transmitted bits. To correspond to the three operations of the channel there are three different types of moves possible on the grid, illustrated in figure 3(b):

Normal transmission (possibly including a bit flip) A diagonal move in which the top left-hand end of the move lies on the intersection between the row and column corresponding to the transmitted and received bits respectively.

Insertion This corresponds to a horizontal move in which an extra random bit appears in the received stream at the left hand end of the move.

Deletion This is a vertical move such that the transmitted bit corresponding to the top of the move does not have a position in the received stream.

We assume prior probabilities, $g_j = \Pr(t_j = 1)$, on each transmitted bit, t_j . The probability of a particular path and set of received of data \mathbf{r} is:

$$\Pr(\text{path}, \mathbf{r}|\mathbf{g}) = \Pr(\text{path}) \cdot \Pr(\mathbf{r}|\text{path}, \mathbf{g}) \quad (1)$$

$$\begin{aligned}
&= \prod_{\text{insertions}} p_{\text{ins}} \prod_{\text{deletions}} p_{\text{del}} \prod_{\text{normal}} (1 - p_{\text{del}} - p_{\text{ins}}) \cdot \\
&\quad \prod_{\text{insertions}} \frac{1}{2} \prod_{\text{normal}} \kappa(r_i, g_j)
\end{aligned} \tag{2}$$

where

$$\kappa(r, g) = \Pr(r|g, \text{normal transmission}, p_{\text{flip}}) \tag{3}$$

For non-iterative resynchronization g_j is 0 or 1 in a marker and $\frac{1}{2}$ otherwise. In iterative resynchronization these “prior” probabilities outside the markers are the messages passed back from the outer low-density parity-check code. The probability of the path is given by a product of the probabilities for each insertion (p_{ins}), deletion (p_{del}) and normal transmission. The probability of the received data for an inserted bit is $\frac{1}{2}$ (in our channel model it is equally likely to be a 0 or 1). Deletions do not give any received data and hence do not contribute towards the probability of the received data.

When decoding we use the forward-backward algorithm [8] to marginalize across all paths. We define a forward probability at a position (i, j) :

$$p_f(i, j) = \Pr(\text{path goes through } (i, j), r_1 \dots r_{i-1} | \mathbf{g}) \tag{4}$$

There are only three ways the path can get to (i, j) – it can arrive by a horizontal, vertical or diagonal move from an adjacent space. So $p_f(i, j)$ can be found recursively:

$$p_f(i, j) = p_{\text{ins}} \frac{1}{2} p_f(i-1, j) + p_{\text{del}} p_f(i, j-1) + (1 - p_{\text{del}} - p_{\text{ins}}) \kappa(r_{i-1}, g_{j-1}) p_f(i-1, j-1) \tag{5}$$

as shown in figure 4. The boundary conditions, assuming that the synchronization error is known to be zero initially, are $p_f(0, 0) = 1$, and $p_f(i, j) = 0$ for all points not on the grid.

Similarly, we define the backward probabilities:

$$p_b(i, j) = \Pr(r_i \dots r_N | \text{path through } (i, j)) \tag{6}$$

The backward message-passing algorithm for computing $p_b(i, j)$ is similar to the forward algorithm above. The boundary conditions are $p_b(i, j) = 0$

for (i, j) not on the grid, and $p_b(N, i) = 1$ for all rows i , where N is the last column of received data.

The forward and backward messages p_f and p_b are used to infer the posterior probability of each user bit, $f_j = \Pr(t_j|\mathbf{r})$, by marginalizing over the diagonal and vertical edges in row j .

3 Comparison with Watermark Codes

In [17] it was shown that for a code system like that in figure 2 the capacity of the effective channel seen by an outer code is higher for marker codes than for watermark codes at low noise levels. The highest rate results published for watermark codes [6] are at $R = 0.71$ with a block size of 4995. To match this a marker code of the same overall rate and block size was created, code A. The inner code was chosen to be good at a noise level $p_{\text{ins}} = p_{\text{del}} = 0.005$ using the effective capacity technique outlined in [17]. The outer code was a low-density parity-check code [14] with weight-2 and weight-3 columns. The code parameters are shown in table 1. Decoding was as figure 2.

A comparison of the watermark and marker codes is shown in figure 5. The marker code outperforms the watermark code, despite the watermark code having been constructed with a code defined over a larger field (GF(16)) than the binary codes considered here.

The performance of the marker code is not close to the channel capacity bounds. To try to approach the bounds, the outer code was optimized. The threshold of an infinite loop-free low-density parity-check code as a function of column weight distribution was evaluated using a Monte Carlo approach [5]. In this procedure it is necessary to know the distribution of messages sent by the channel. Statistics of the messages received by the outer code were collected, figure 6. The distribution is almost symmetrical if given a data sequence of i.i.d. bits. The simulation was carried out with an all zero-transmission but with noise statistics as if they were i.i.d. data bits.

Despite the non-Gaussian form of the messages, degree sequences could not be found that significantly outperformed good degree sequences obtained from optimizations on the Gaussian channel [4]. This optimization was carried out with the message histograms marginalized over all bits, figure 6(b). Perhaps if a different degree sequence for each bit between markers were allowed, a further gain could be achieved; as figure 6(a) shows, bits received near markers are more reliable than bits far away from markers.

Simulations using a good degree sequence from the Gaussian channel, code B, are also shown in figure 5. The waterfall region is not much closer to the channel capacity bounds, but it is close to the serial “capacity” (defined from the effective capacity of the channel seen by the outer code [17]).

4 A Complete Iterative System

The outer low-density parity-check code is decoded with loopy belief propagation. Various papers (for example [15, 25]) have shown that extending this loopy belief propagation to include channel state estimation can be beneficial to decoding performance. For the insertion-deletion resynchronization phase it is expected that if we give the resynchronization algorithm more information about the likely transmission, the accuracy of the resynchronization will increase.

Watermark and regular marker codes were empirically discovered to have similar performance near $R = 0.5$ by looking at the capacity of the effective channel [17]. Therefore $R = 0.5$ codes with a blocksize of 4000 (to match [6]) were studied.

As benchmarks, marker codes were decoded with the serial decoding algorithm from section 3. The first simulation (code C) was with identical markers of length 3 and a low-density parity-check outer code with weight-2 and weight-3 columns. Figure 7 shows that the watermark code has better error floor behaviour. With identical markers catastrophic error propaga-

tion is possible in a marker code as the resynchronization can be shifted by a multiple of a marker interval. Code D was similar to code C however each marker was pseudo-randomly chosen from a set of two different markers. This improved the error floor and led to better performance than the watermark code. An irregular low-density parity-check outer code (chosen to be good on the Gaussian channel) was also tested and a small further improvement found, code E.

Simulations were carried out with codes D and E using iterative resynchronization. The algorithm is similar to the serial resynchronization algorithm but with extrinsic information from the low-density parity-check decoder fed back into the resynchronization stage (updating the values for \mathbf{g} in equation 2). The algorithm can be seen as loopy belief propagation on a factor graph like figure 8. The resynchronization was carried out every five iterations of the low-density parity-check decoder to increase the decoding speed as a low-density parity-check decoder iteration is faster than probabilistic resynchronization. A more efficient schedule of steps may be possible [2].

The simulation results are shown on figure 7. The figure shows that the iterative approach significantly outperforms the serial approach and that the waterfall region can be close to the channel capacity. It is worth noting that the ranking of the codes is reversed between serial and iterative decoding. This suggests that the choice of code to be used should be made in conjunction with the decoding algorithm.

5 EXIT charts

To look into this swap in performance between serial and iterative decoding, extrinsic information transfer (EXIT) charts [23] for the system were studied at a noise level of $p_{\text{ins}} = p_{\text{del}} = 0.04$. EXIT charts allow a visualization of the messages passed during decoding between two loop-free sections of the graph

describing the decoding. When a section is loop-free an “exact” inference can be carried out in that subsection of the graph. At the limit of large block size we can then look at the average statistical properties of the messages into that section versus the messages coming out of that section. We assume a Gaussian form to the distribution of messages. The transfer function of the two graph sections can be put on opposing axes and then the expected progress of infinite-blocksize decoding can be seen, as shown in figure 9. A “staircase” is formed between the two curves with a step per iteration. To decode, the staircase needs to reach the top right-hand corner (which indicates totally confident messages). If there is an intersection between the two curves this is not possible and a code is not expected to decode.

The EXIT chart of the resynchronization was obtained by a Monte Carlo approach and a quadratic function fitted, figure 10(a). With no information passed from the code to the resynchronization stage the maximum rate of the outer code with serial decoding is shown at the intercept with the y -axis. As the input information is increased the output information increases but does not reach 1. This is due to remaining uncertainty in the exact synchronization path and whether the bit in question may have been deleted (and possibly reinserted).

The behaviour of the decoding algorithm can be seen in terms of messages being passed between the variable nodes and check nodes; a section made up of check nodes and a section made up of variable nodes and resynchronization are each individually loop-free. The transfer functions of the check and variable nodes were calculated using the approximate forms from [23]. We plot the complete EXIT charts in terms of these two loop-free sections. When the transfer function of the resynchronization is combined with the transfer function of the variable nodes, the resultant transfer function is expected to reach the top-right corner of the EXIT chart as the variable nodes have degree greater than 1.

The code that performs well on the Gaussian channel leads to an EXIT chart with an intersection, figure 10(b), so decoding is not expected to converge. For the code with only weight-2 and weight-3 columns no intersection is observed, figure 10(c), and therefore decoding is expected to converge. A code with a softer response often performs better as part of an iterative decoding scheme (for an extreme example see [16]).

The width of the swath between the curves can be used as a metric to choose a better degree sequence. We only have the resynchronization EXIT chart at one noise level. To create a better code we want to find a form of curves that is less likely to have an intersection at a higher noise level. As the noise increases the top curve moves down the chart, so keeping a wide swath between the curves allows a larger noise level to be reached until an intersection occurs. Better degree sequences were searched for using a global optimization package [10]. The function maximized was the scale factor applied to the top curve from $y = 1$ such that the scaled top curve and bottom curve touch without crossing. Better thresholds could only be found by increasing the number of weight-2 columns above the number of rows in the parity-check matrix. This is not expected to produce a good code as short cycles in weight-2 columns lead to low weight codewords.

6 Deletion Channel

Interest in the academic community has recently focused on the bit-deletion channel (insertion-deletion channels where $p_{\text{ins}} = 0$). Researchers have generally used similar codes to those used on the insertion-deletion channel [19].

Simulations of the marker codes developed in this paper for the insertion-deletion channel are shown on the bit-deletion channel in figure 11. The codes significantly outperform other known deletion codes (for example allowing approximately twice the transmission rate of [3]).

7 Conclusion

We have shown that marker codes outperform watermark codes at rates above 0.5. To avoid catastrophic decoding errors at higher noise levels the use of pseudo-random markers from a set of different markers is necessary.

Iterative resynchronization provides better performance than serial resynchronization, bringing the waterfall region close to the bounds on the channel capacity.

A Biography

Edward A. Ratzler has recently completed his PhD (Error-Correcting Communication on Non-Standard Channels) under the supervision of David J.C. MacKay at the University of Cambridge.

References

- [1] P.A.H. Bours. *Codes for Correcting Insertion and Deletions Errors*. PhD thesis, Eindhoven Technical University, June 1994.
- [2] Fredrik Brännström, Lars K. Rasmussen, and Alex Grant. Optimal scheduling for iterative decoding. In *International Symposium on Information Theory*, page 350, July 2003.
- [3] Johnny Chen, Michael Mitzenmacher, Chaki Ng, and Nedeljko Varinica. Concatenated codes for deletion channels. In *IEEE International Symposium on Information Theory*, page 218, July 2003.
- [4] Sae-Young Chung. LDPC code design applet. On the web at <http://lids.mit.edu/~sychung/gaopt.html>.

- [5] Matthew C. Davey. *Error-Correction Using Low-Density Parity-Check Codes*. PhD thesis, University of Cambridge, 1999. Available at <http://www.inference.phy.cam.ac.uk/mcdavey>.
- [6] Matthew C. Davey and David J.C. MacKay. Reliable communication over channels with insertions, deletions, and substitutions. *IEEE Transactions on Information Theory*, 47(2):687–698, February 2001.
- [7] M.P.F. dos Santos, W.A. Clarke, H.C. Ferreira, and T.G. Swart. Correction of insertions/deletions using standard convolutional codes and the Viterbi decoding algorithm. In *IEEE Information Theory Workshop*, pages 187–190, April 2003.
- [8] Richard Durbin, Sean R. Eddy, Anders Krogh, and Graeme Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge University Press, 1998.
- [9] H.C. Ferreira, W.A. Clarke, A.S.J. Helberg, K.A.S. Abdel-Ghaffar, and A.J. Han Vinck. Insertion/deletion correction with spectral nulls. *IEEE Transactions on Information Theory*, 43(2):722–732, March 1997.
- [10] Joerg M. Gablonsky and C. Tim Kelley. A locally-biased form of the DIRECT algorithm. *Journal of Global Optimization*, 21:27–37, 2001.
- [11] Frederick F. Sellers Jr. Bit loss and gain correction code. *IEEE Transactions on Information Theory*, 8(1):35–38, January 1962.
- [12] Stavros Konstantinidis, Steven Perron, and L. Amber Wilcox-O’Hearn. On a simple method for detecting synchronization errors in coded messages. *IEEE Transactions on Information Theory*, 49(5):1355–1363, May 2003.

- [13] V.I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics – Doklady*, 10(8):707–710, February 1966.
- [14] David J.C. MacKay. Good error correcting codes based on very sparse matrices. *IEEE Transactions on Information Theory*, 45(2):399–431, 1999.
- [15] Edward A. Ratzler. Low-density parity-check codes on Markov channels. In *Second IMA Conference on Mathematics in Communications*, December 2002.
- [16] Edward A. Ratzler. Intersected low-density parity-check and convolutional codes. In *7th International Symposium on Communication Theory and Applications*, July 2003.
- [17] Edward A. Ratzler and David J.C. MacKay. Codes for channels with insertions, deletions and substitutions. In *2nd International Symposium on Turbo Codes and Related Topics*, 2000.
- [18] Leonard J. Schulman and David Zuckerman. Asymptotically good codes correcting insertions, deletions and transpositions. *IEEE Transactions on Information Theory*, 45(7):2552–2557, November 1999.
- [19] Neil J. Sloane. On single-deletion-correcting codes. In K.T. Arasu and A. Seress, editors, *Codes and Designs, Ohio State University, May 2000*, pages 273–291, 2002.
- [20] J.J. Stiffler. *Theory of synchronous communications*. Prentice-Hall, 1971.
- [21] L.R. Welch S.W. Golomb, B. Gordon. Comma free codes. *Canadian Journal of Mathematics*, 10(2):202–209, 1958.

- [22] T.G. Swart and H.C. Ferreira. Insertion/deletion correcting coding schemes based on convolution coding. *Electronics Letters*, 38(16):871–873, August 2002.
- [23] Stephan ten Brink, Gerhard Kramer, and Alexei Ashikhmin. Design of low-density parity-check codes for multi-antenna modulation and detection. Submitted to *IEEE Transactions on Communications*, June 2002.
- [24] Jeffrey D. Ullman. On the capabilities of codes to correct synchronization errors. *IEEE Transactions on Information Theory*, 13(1):95–105, January 1967.
- [25] Andrew P. Worthen and Wayne E. Stark. Unified design of interactive receivers using factor graphs. *IEEE Transactions on Information Theory*, 47(2):843–849, February 2001.
- [26] Kamil Sh. Zigangirov. Sequential decoding for a binary channel with drop-outs and insertions. *Problemy Peredachi Informatsii*, 5(2):23–30, 1969.

List of Figures

1	Flow chart (from [17]) describing the binary insertion-deletion channel with insertion probability p_{ins} , deletion probability p_{del} , transmission probability $p_t = 1 - p_{\text{ins}} - p_{\text{del}}$, and substitution probability p_{flip} . For simplicity $p_{\text{flip}} = 0$ for the simulations in this paper.	16
2	The code system used, with serial decoding illustrated	17
3	Synchronization represented by a path on a two dimensional grid	18
4	The recursive evaluation of the forward probabilities	19
5	$R = 0.71$, $N = 4995$ codes over an insertion/deletion channel with serial decoding. The watermark code result is from [6]. Marker code A is with a low-density parity-check code with with weight-2 and weight-3 columns. Marker code B has weight 10 columns in addition, table 1.	20
6	Histograms of the probability density functions of messages received by an outer code where the inner code has a marker sequence of ‘01’ between every 19 data bits. The log-likelihood ratio (LLR) is $\frac{\Pr(\text{bit}=1)}{\Pr(\text{bit}=0)}$	21
7	The benefit of full iterative decoding with $R = 0.5$, $N = 4000$ codes. Identical marker codes are shown simulated with iterative and non-iterative resynchronization. Also the reduction in error floor from identical markers (C) to non-identical markers (D) is shown. The outer codes in codes C and D have only weight 2 and 3 columns, code E has weight 10 columns in addition. The watermark code result is from [6] and the details of the marker codes are in table 1.	22
8	A factor graph of the coding system. Iterative decoding can be seen as a loopy belief propagation algorithm on the factor graph. The region where the \mathbf{f} and \mathbf{g} messages of section 2 are passed is shown.	23
9	An example of an EXIT chart of a graph split into two sections (“right” and “left”). A condition where decoding is expected to converge is shown with a schematic progress of decoding shown with a thick line.	24
10	EXIT charts at $p_{\text{ins}} = p_{\text{del}} = 0.04$	25
11	Codes of $R = 0.5$ and $R = 0.71$ simulated on the deletion channel ($p_{\text{ins}} = 0$). The lower bounds on the Shannon Limit for $R = 0.71$ and $R = 0.5$ are $p_{\text{del}} = 0.0509$ and $p_{\text{del}} = 0.110$ respectively.	26

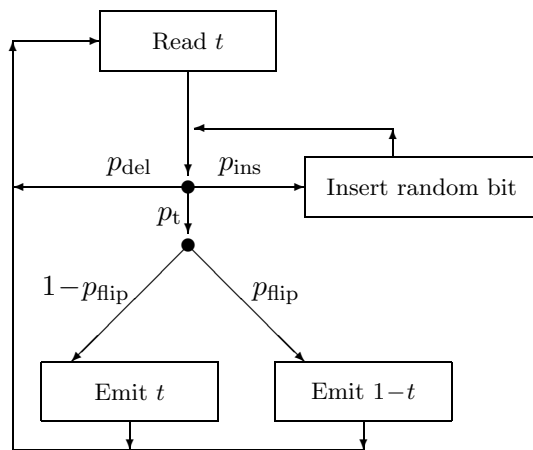


Figure 1: Flow chart (from [17]) describing the binary insertion-deletion channel with insertion probability p_{ins} , deletion probability p_{del} , transmission probability $p_t = 1 - p_{\text{ins}} - p_{\text{del}}$, and substitution probability p_{flip} . For simplicity $p_{\text{flip}} = 0$ for the simulations in this paper.

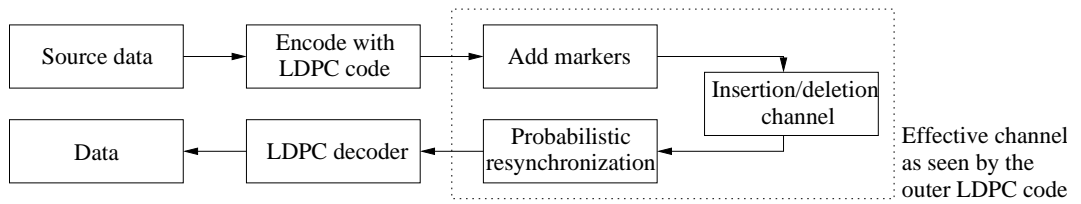
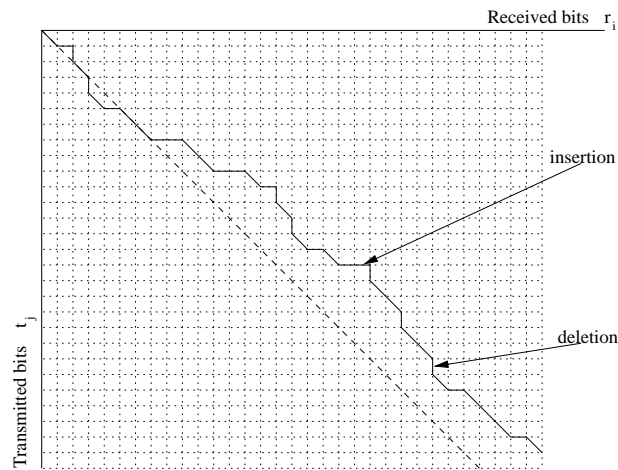
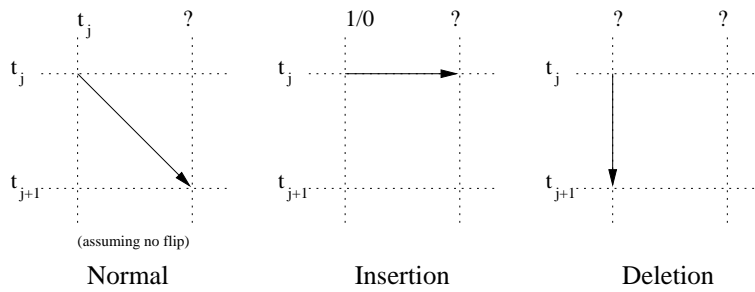


Figure 2: The code system used, with serial decoding illustrated



(a) An example path



(b) The allowed moves

Figure 3: Synchronization represented by a path on a two dimensional grid

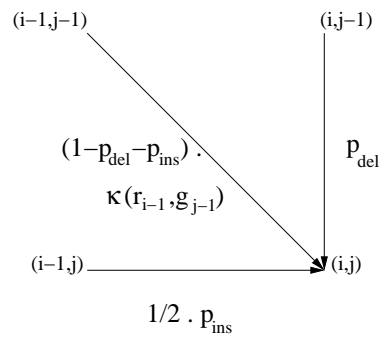


Figure 4: The recursive evaluation of the forward probabilities

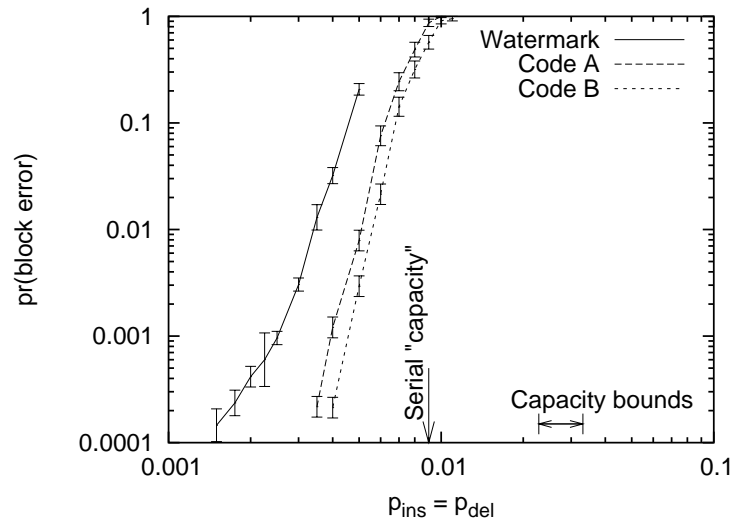
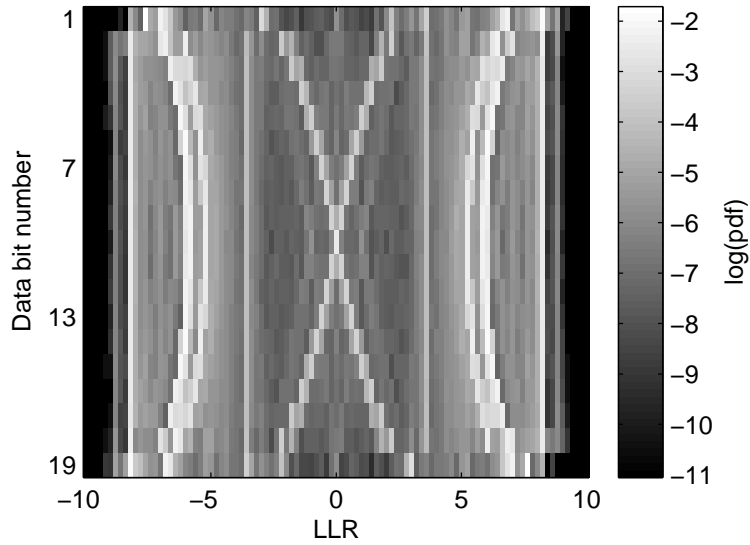
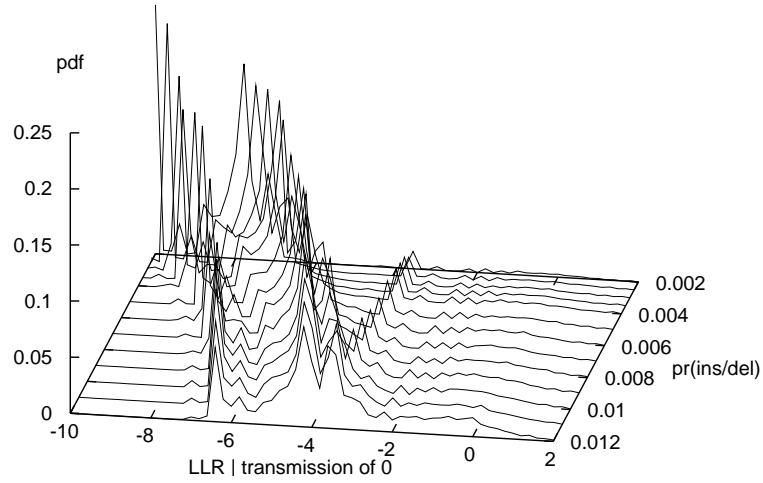


Figure 5: $R = 0.71$, $N = 4995$ codes over an insertion/deletion channel with serial decoding. The watermark code result is from [6]. Marker code A is with a low-density parity-check code with with weight-2 and weight-3 columns. Marker code B has weight 10 columns in addition, table 1.



(a) At a noise level of $p_{\text{ins}} = p_{\text{del}} = 0.5\%$. The vertical axis ranges over each data bit between two markers. Close to a marker the messages are more confident as a larger value of the absolute LLR is more likely.



(b) Marginalized over all data bits whose transmitted bit was '0'

Figure 6: Histograms of the probability density functions of messages received by an outer code where the inner code has a marker sequence of '01' between every 19 data bits. The log-likelihood ratio (LLR) is $\frac{\Pr(\text{bit}=1)}{\Pr(\text{bit}=0)}$.

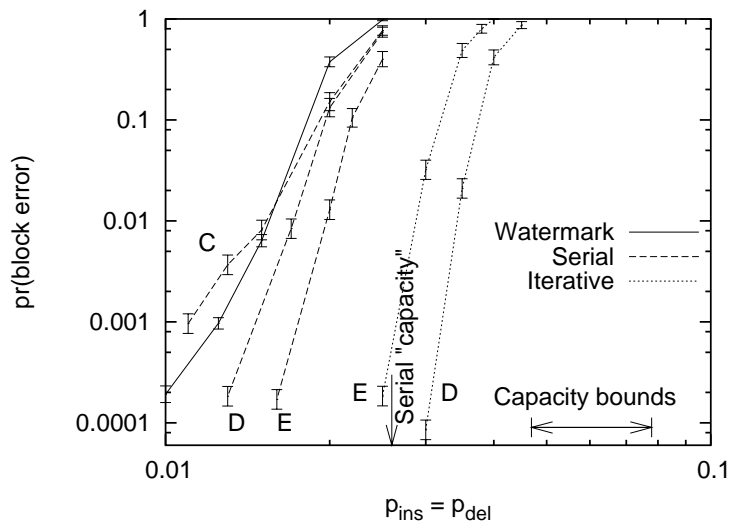


Figure 7: The benefit of full iterative decoding with $R = 0.5$, $N = 4000$ codes. Identical marker codes are shown simulated with iterative and non-iterative resynchronization. Also the reduction in error floor from identical markers (C) to non-identical markers (D) is shown. The outer codes in codes C and D have only weight 2 and 3 columns, code E has weight 10 columns in addition. The watermark code result is from [6] and the details of the marker codes are in table 1.

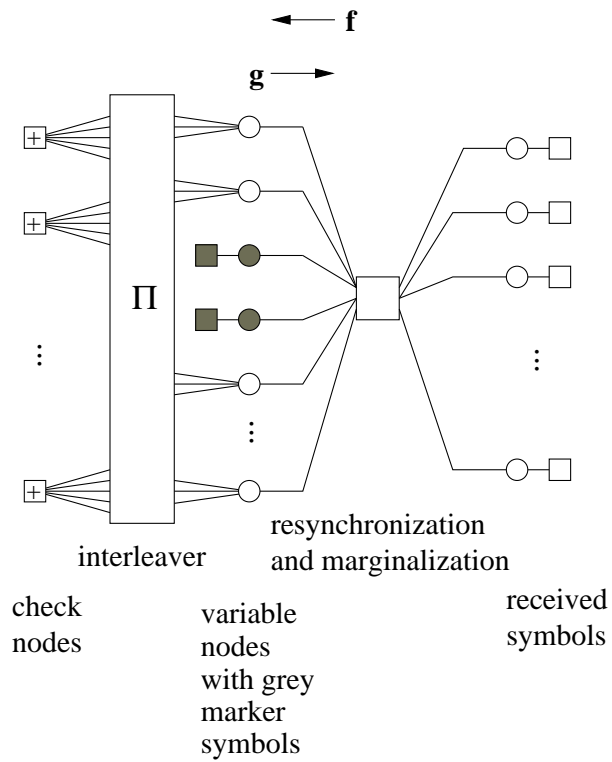


Figure 8: A factor graph of the coding system. Iterative decoding can be seen as a loopy belief propagation algorithm on the factor graph. The region where the \mathbf{f} and \mathbf{g} messages of section 2 are passed is shown.

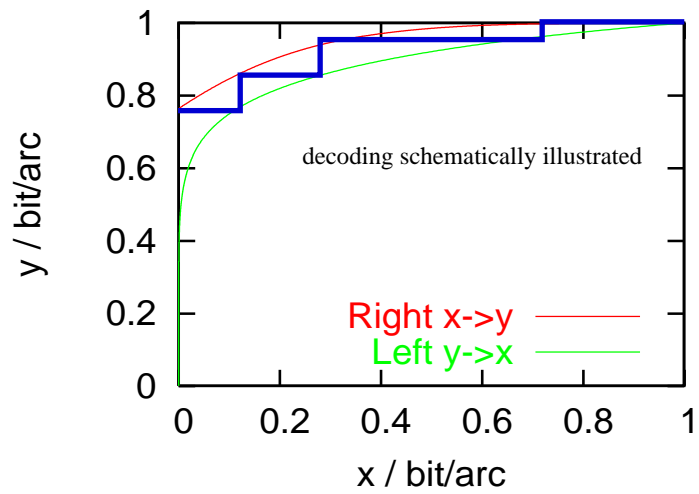
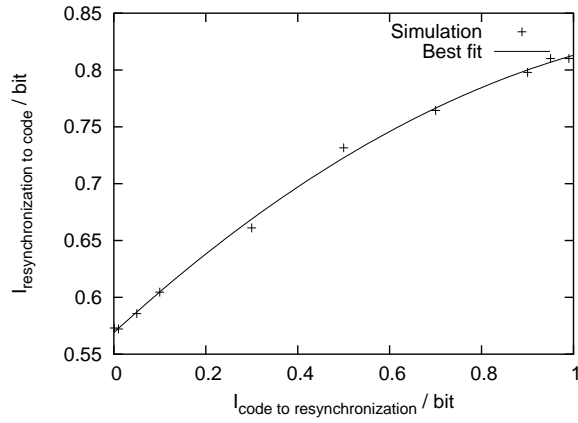
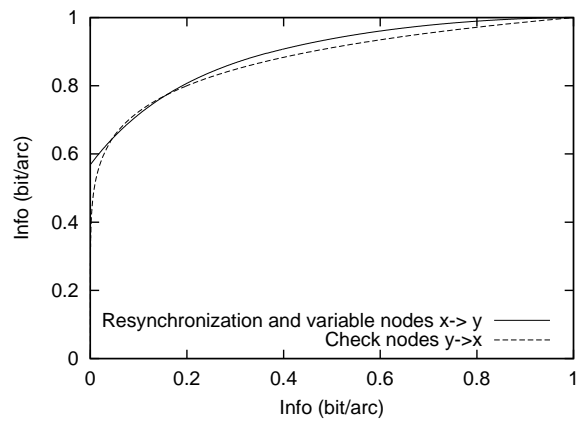


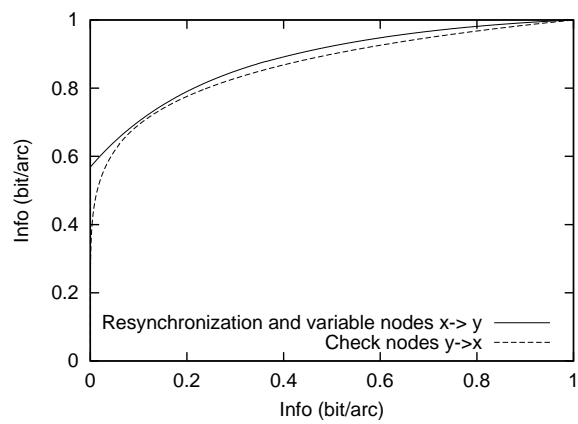
Figure 9: An example of an EXIT chart of a graph split into two sections (“right” and “left”). A condition where decoding is expected to converge is shown with a schematic progress of decoding shown with a thick line.



(a) Insertion deletion channel



(b) Code E



(c) Code D

25
Figure 10: EXIT charts at $p_{\text{ins}} = p_{\text{del}} = 0.04$

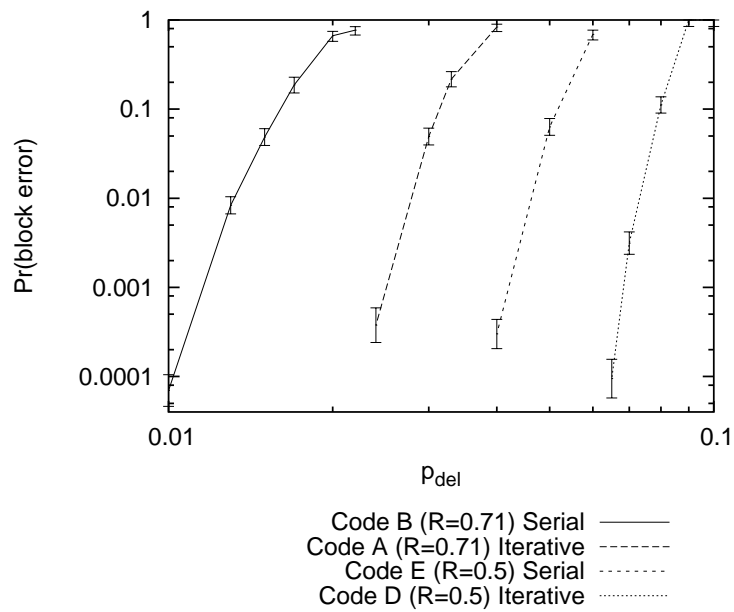


Figure 11: Codes of $R = 0.5$ and $R = 0.71$ simulated on the deletion channel ($p_{\text{ins}} = 0$). The lower bounds on the Shannon Limit for $R = 0.71$ and $R = 0.5$ are $p_{\text{del}} = 0.0509$ and $p_{\text{del}} = 0.110$ respectively.

List of Tables

- 1 The codes used in this paper. Markers (m) are inserted between every d data bits. If more than one marker is available, the marker is chosen pseudo-randomly. The low-density parity-check outer code was constructed with c_i columns of weight i 28

	A	B	C	D	E
R	0.71	0.71	0.5	0.5	0.5
N	4995	4995	4000	4000	4000
d	19	19	9	9	9
m	01	01	001	001/110	001/110
N_{LDPC}	4521	4521	3001	3001	3001
M_{LDPC}	969	969	1001	1001	1001
c_2	968	963	1000	1000	927
c_3	3553	2785	2001	2001	1572
c_{10}	0	773	0	0	502

Table 1: The codes used in this paper. Markers (m) are inserted between every d data bits. If more than one marker is available, the marker is chosen pseudo-randomly. The low-density parity-check outer code was constructed with c_i columns of weight i .