A New Data Compression Algorithm for Sources with Memory based on Error Correcting Codes

Giuseppe Caire
Institut Eurecom
giuseppe.caire@eurecom.fr

Shlomo Shamai
Technion
sshlomo@ee.technion.ac.il

Sergio Verdú
Princeton University
verdu@princeton.edu

Abstract — **A new fixed-length asymptotically optimal scheme for lossless compression of stationary ergodic tree sources with memory is proposed. Our scheme is based on the concatenation of the Burrows-Wheeler block sorting transform with the syndrome former of a linear error-correcting code. Low-density parity-check (LDPC) codes together with Belief Propagation decoding lead to linear compression and decompression times, and to natural universal implementation of the algorithm.**

I. Introduction

We propose a new approach to lossless data compression based on error correcting codes and the block-sorting transform. Existing zero-error variable-length data compression algorithms suffer from the following shortcomings when used in data-transmission applications through noisy channels:

- Lack of resilience to transmission errors.
- Packetized transmission/recording often means that the compression output length is only relevant up to an integer multiple of a given packet length.
- The asymptotic regime for which the algorithms approach the source entropy rate requires much longer packet lengths than those use in modern high-speed wireless applications.
- The traditional separation approach to source-channel coding enforces the artificial constraint that errors are uniquely due to the channel and does not lend itself naturally to packetized transmission.
- Incorporation of prior knowledge of source statistics is cumbersome for some lossless algorithms.
- Zero-error variable-length data compression algorithms cannot be adapted for use in Slepian-Wolf separate encoding of correlated sources.

In this paper we show that our approach is quite competitive with existing algorithms even in the purely lossless data compression setting. The gains over the conventional schemes in the setting of source-channel coding will be reported elsewhere.

II. Memoryless Sources

The interplay of error correcting codes and noiseless data compression has received scant attention in the literature. Dealing with source sequences known to have a limited Hamming weight, Weiss [18] noted that since the 2^{n-k} cosets of an e-correcting (n, k) binary linear code can be represented by their unique codewords having weight less than or equal to e, the binary input n-string can be encoded with $n - k$ bits that label the coset it leads. The decompressor is simply the mapping that identifies each coset with its leader. The fact that this is equivalent to computing the syndrome that would result if the source sequence were the channel output was explicitly recognized by Allard and Bridgewater [1]. Both [1] and Fung et al [9] remove Weiss' bounded weight restriction and come up with variable-to-fixed-length schemes where zeros are appended prior to syndrome formation so that the resulting syndrome is correctable. Ancheta [2] considered fixed-to-fixed linear source codes based on syndrome formation and dealt with binary memoryless sources and a bit-error-rate performance measure, rather than a "noiseless" or "almost-noiseless" approach. Since the only sources the approach in [18, 1, 9, 2] was designed to handle were memoryless sources with known distributions (essentially biased coins) and since the practical error correcting codes known at that time did not offer rates close to channel capacity, that line of research came to a close. Also limited to memoryless binary sources is another, more recent, approach [12] in which the source is encoded by a Turbo code whose systematic outputs as well as a portion of the nonsystematic bits are discarded; and decompression is carried out by a Turbo decoder that uses knowledge of the source bias. Also germane to this line of research is the use of Turbo and LDPC codes [3, 11, 15, 14] for Slepian-Wolf coding of correlated memoryless sources as specific embodiments of the approach proposed in [19].

Linear source codes are known to achieve the entropy rate of memoryless sources [5]. We have extended this result to very general sources with memory and nonstationarity. If $\mathbf{s}$ is the source vector, given the linearly encoded vector $\mathbf{Hs}$ the maximum likelihood decoder selects $g_{\mathbf{H}}(\mathbf{Hs})$, the most likely source vector $\mathbf{u}$ that satisfies $\mathbf{Hu} = \mathbf{Hs}$. On the other hand, consider an additive-noise discrete channel $\mathbf{y} = \mathbf{x} + \mathbf{u}$ driven by a linear channel code with parity check matrix $\mathbf{H}$. It is easy to see that the maximum likelihood decoder selects the codeword $\mathbf{y} - g_{\mathbf{H}}(\mathbf{Hy})$, i. e. it selects the most likely noise realization among those that lead to the same *syndrome* $\mathbf{Hy}$. Thus, the problems of almost-noiseless fixed-length data compression and almost noiseless coding of an additive-noise discrete channel whose noise has the same statistics as the source are identical. This strongly suggests using the parity-check matrix of a channel code as the source encoding matrix. One of the immediate (but neglected) consequences of this powerful equivalence relates to the construction of optimum n-to-m source codes, which are well-known to be those that give a

0-7803-7799-0/03/$17.00 © 2003 IEEE

unique codeword to the n-sequences with the largest probabilities. However, even for biased coins no explicit constructions have previously been given. This can be accomplished for certain (m, n) by using the parity-check matrix of perfect codes. For example, in the binary case, we can obtain optimum 7-to-4 and 23-to-12 codes by using the Hamming and Golay codes respectively.

For general good codes, syndrome forming (source encoding) has quadratic complexity in the blocklength and maximum likelihood decoding has exponential complexity in the blocklength. When the code is an LDPC, $\mathbf{H}$ is a sparse matrix with a number of nonzero entries growing linearly with the blocklength, and, thus, the compressor has linear complexity in the blocklength n. Note that this is in contrast to the encoding of LDPC codes for channel encoding as the multiplication by the generator matrix is far less straightforward, requiring the application of the technique in [17] to achieve nearly linear complexity. The sparseness of $\mathbf{H}$ does not imply that a polynomially-complex maximum likelihood decoder exists. For LDPCs the suboptimal iterative technique known as *Belief-Propagation* decoding (BP) has proved to yield very good results on a variety of memoryless channels. The BP decoders used in channel decoding of binary symmetric channels [10, 16] operate with the channel outputs rather than with the syndrome vector. Thus, they are not directly useful for our purposes. However, counterparts to those algorithms that give an approximation to the syndrome-decoding function $g_{\mathbf{H}}$ can be found. In these algorithms, the data is not present at the variable nodes (which represent the uncompressed bits) but at the check nodes, since in the source-coding case, each parity-check equation has a value given by the compressed data. For simplicity and concreteness we specify the algorithm in the binary memoryless case with nonstationary probabilities (which will be relevant in Section III). Fix the realization of the input to the decoder, $\mathbf{z}$. The set of checknodes in which the bitnode $k \in \{1, \dots n\}$ participates is denoted by $\mathcal{A}_k \subset \{1, \dots m\}$, and the set of bitnodes which are connected to checknode $j \in \{1, \dots m\}$ is denoted by $\mathcal{B}_j \subset \{1, \dots n\}$. Define the a priori source log-ratios

$$\mathcal{L}_k = \log \frac{1 - p_k}{p_k}, \quad k \in \{1, \dots n\} \tag{1}$$

For each iteration $t = 1, 2, \dots$, the algorithm computes the value of the bitnodes

$$\widehat{x}_k = \operatorname{sign}\left\{ \mathcal{L}_k + \sum_{j \in \mathcal{A}_k} \mu^{(t)}_{j \to k} \right\}$$

by updating the messages sent by the checknodes to their neighboring bitnodes and by the bitnodes to their neighboring checknodes, denoted respectively by $\mu^{(t)}_{j \to k}$ and by $\nu^{(t)}_{k \to j}$, according to the message-passing rules

$$\nu^{(t)}_{k \to j} = \mathcal{L}_k + \sum_{j' \in \mathcal{A}_k - \{j\}} \mu^{(t-1)}_{j' \to k} \tag{2}$$

and

$$\mu^{(t)}_{j \to k} = (-1)^{z_j} 2 \tanh^{-1} \left(\prod_{k' \in \mathcal{B}_j - \{k\}} \tanh(\nu^{(t)}_{k' \to j}/2) \right) \tag{3}$$

with the initialization $\mu^{(0)}_{j \to k} = 0$ for all $j \in \{1, \dots, m\}$. At each iteration, the parity-check equations

$$\sum_{k \in \mathcal{B}_j} \widehat{x}_k = z_k, \quad j = 1, \dots, m$$

are evaluated. If they are all satisfied, the algorithm stops. If after a maximum number of allowed iterations some parity-check equations are not satisfied, a block-error is declared.

By using the symmetries of bitnode and checknode mappings, we have shown that the same pointwise correspondence between channel decoding and source decoding stated for the ML decoder holds also for the BP decoder: the sets of source sequences and noise sequences that lead to errors of the respective BP decoders are identical.

In contrast to channel decoding, in noiseless data compression, the source encoder has the luxury of running an exact copy of the decoder iterations. We have taken advantage of this fact to design an algorithm that noticeably decreases the block error probability of the scheme. Our *Iterative Doping Algorithm* makes use of multiple codebooks and symbol doping, which proceeds along the BP iterations at the cost of one bit per iteration. For a given parity-check matrix $\mathbf{H}$ and source sequence $\mathbf{s}$ the syndrome $\mathbf{z} = \mathbf{Hs}$ is computed. Then, the position of the doped symbols is computed iteratively as follows:

1) Initialization: $\mu^{(0)}_{j \to k} = 0$ for all $j \in \{1, \dots, m\}$.

2) Repeat the following steps for $t = 1, 2, \dots$ until successful decoding is reached:

- Reliability sorting: for all bitnodes $k = 1, \dots, n$ compute

$$\nu^{(t)}_k = \mathcal{L}_k + \sum_{j \in \mathcal{A}_k} \mu^{(t-1)}_{j \to k}$$

  and sort the values $|\nu^{(t)}_k|$ in increasing order.

- Least-reliable symbol doping: let

$$\widehat{k} = \arg \min_{k = 1, \dots, n} \{|\nu^{(t)}_k|\},$$

  feed the symbol $x_{\widehat{k}}$ directly to the decoder and let

$$\mathcal{L}_{\widehat{k}} = \begin{cases} +\infty & \text{if } x_{\widehat{k}} = 0 \\ -\infty & \text{if } x_{\widehat{k}} = 1 \end{cases}$$

- Bitnode update (2) and Checknode update (3).

At each decoder iteration, the source symbol for which the BP algorithm has accumulated the least reliability is communicated to the decoder. Some qualitative properties of the Iterative Doping Algorithm are: 1) Because of reliability sorting, the position of doped symbols need not be explicitly communicated to the decoder. Hence, the cost of doping is the (random) number of iterations necessary to achieve successful decoding. 2) The algorithm never dopes twice the same symbol. 3) The algorithm stops in at most n iterations, i.e., it never expands the source sequence length. 4) Doping the symbol with least reliability is an effective strategy. In fact, subject to the assumption of cycle-free graph, at each iteration t the conditional entropy of each symbol given the messages

0-7803-7799-0/03/$17.00 © 2003 IEEE

in the oriented neighborhood of depth $2t$ decreases with the symbol reliability. Since doping a symbol costs one bit, it is intuitively convenient to spend this bit to correct the symbol with the largest conditional entropy at each iteration.

The iterative doping algorithm implements a variable-length scheme. The mean and variance of the number of doped symbols can be reduced by using a library of several parity-check matrices, by applying the above algorithm to each matrix, choosing the one that achieves successful decoding with the smallest length, and indicating the label of the best matrix to the decoder.

III. Sources with Memory

So far we have limited our discussion to encoding memoryless nonstationary sources. As we saw, it is easy to incorporate the knowledge about the source marginals in the BP decoding algorithm. However, for sources with memory the marginals alone do not suffice for efficient data compression. It would be futile to search for encoders and decoders as a function of the memory structure of the code. We next describe a design approach that enables the use of a BP (marginal based) decoding algorithm while taking full advantage of the source memory.

We propose to use a one-to-one transformation, called the block-sorting transform or Burrows-Wheeler transform (BWT) [4] which performs the following operation: after adding a special End-of-file symbol, it generates all cyclic shifts of the given data string and sorts them lexicographically. The last column of the resulting matrix is the BWT output from which the original data string can be recovered.

Note that the BWT performs no compression. Fashionable universal data compression algorithms (e.g. *bzip*) have been proposed which are quite competitive with the Lempel-Ziv algorithm. To understand how this is accomplished, it is best to consider the statistical properties of the output of the BWT. It is shown in [6] that the output of the BWT (as the block-length grows) is asymptotically piecewise i.i.d. For stationary ergodic tree sources the length, location, and distribution of the i.i.d. segments depend on the statistics of the source. The universal BWT-based methods for data compression all hinge on the idea of compression for a memoryless source with an adaptive procedure which learns implicitly the local distribution of the piecewise i.i.d. segments, while forgetting the effect of distant symbols.

Our approach is to let the BWT be the front-end. Then we apply the output of the BWT to the LDPC parity-check matrix, as explained in Section II for memoryless nonstationary sources. Moreover, we found it advantageous to set a threshold $H_{\rm th}$ reasonably close to 1 bit/symbol and feed directly to the output the symbols on segments whose entropy exceeds $H_{\rm th}$.

The decompressor consists of the BP decoder, making use of the prior probabilities p_i. The locations of the transitions between the segments are random, and must be communicated explicitly to the decoder.

Once the string has been decompressed we apply the inverse BWT to recover the original string. We note that the inverse BWT does not degrade gracefully with respect to errors. If only one symbol is in error the output will be seriously erroneous. This is yet another reason to gauge the reliability of the almost-noiseless compressor by the Shannon-type block error rate.

The proof of optimality of the scheme for stationary ergodic sources using capacity-approaching channel codes uses the tools developed in [6].

As in the case of memoryless sources, the computational complexity of both compression and decompression grow linearly with the data size.

Since a fixed-to-fixed scheme in principle fixes the compression rate without regard to the actual source realization, we could think that universality is not possible in this approach. However, recall that even within the "almost-noiseless" framework described above the encoder can try the most ambitious compression rate that guarantees success. No knowledge of the source is required by the BWT. However, we need a robust LDPC design procedure that yields ensembles of codes that perform close to capacity for nonstationary memoryless channels. Since the capacity-achieving distribution is the same (equiprobable) regardless of the noise distribution, Shannon theory [5] guarantees that without knowledge of the channel sequence at the encoder it is possible to attain the channel capacity (average of the maximal single-letter mutual informations) with the same random coding construction that achieves the capacity of the stationary channel.

A key modification in order to make the algorithm universal is to include a move-to-front algorithm (e.g. [7]) between the BWT and the LDPC encoder. Indeed, most of the existing data compressors based on BWT use such a module. At the expense of a slight introduction of dependence, move-to-front serves a smoothing effect: making the marginal distributions more stationary. This has beneficial effects at both the encoder and decoder in universal mode. At the encoder, the nonstationarity (both in range and in dynamics) of the marginals seen by the LDPC is lessened; thus the code need not be as robust as if it were applied directly to the output of the BWT. At the decoder, the richness of parameters that the decoder must adapt to decreases considerably since the marginal distributions are varying much more slowly. Run-length coding is also beneficial in order to avoid very low-entropy segments. The existing universal BWT-based methods for variable-length data compression all hinge on the idea of compression for a memoryless source with an adaptive procedure that learns the local distribution implicitly while forgetting the effect of distant symbols. In our universal implementation the encoder estimates the evolving first-order distribution and communicates a rough piecewise approximation to the decoder using a number of symbols that is a small fraction of the total compressed sequence. After each iteration, the BP decoder refines its estimates of individual probabilities on a segment-by-segment basis by weighing the impact of each tentative decision with the current reliability information. Naturally, such procedures are quite easy to adapt to prior knowledge that may exist from the encoding of the previous data blocks for example. Moreover, the encoder can monitor the statistics from block to block. When it sees that there is a large change in statistics it can alert the decoder to the segments that are most affected, or even back off in rate.

IV. Numerical Experiments

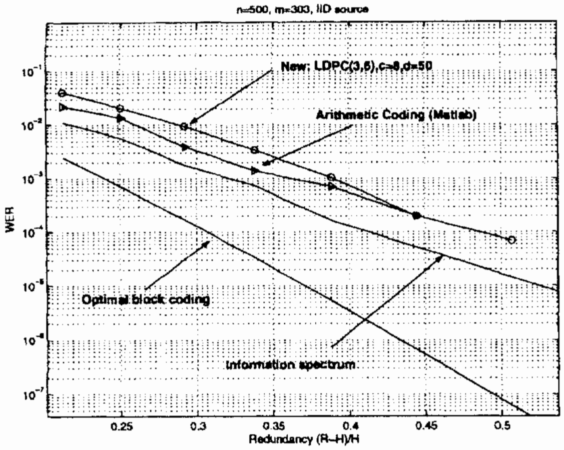

Figure 1: Block error rate of 500-to-303 source codes for a biased coin.

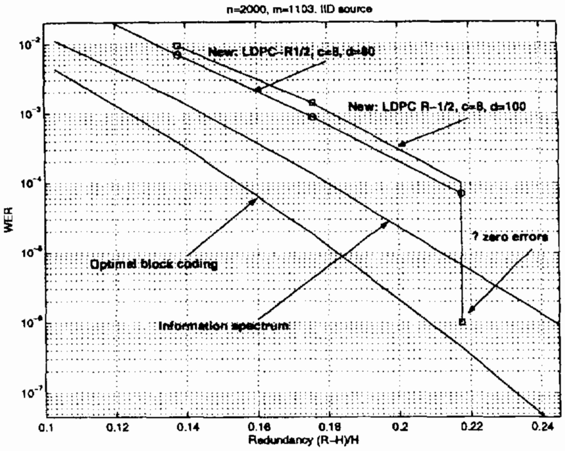

Figure 2: Block error rate of 2000-to-1103 source codes for a biased coin.

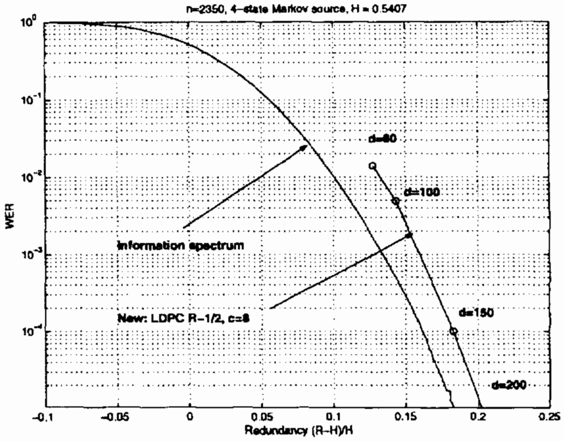

Figure 3: Block error rate of a 2350-to-(1350+d) source code for a four-state Markov chain.

In this section we test the nonuniversal version of our algorithm for binary memoryless sources and simple Markov chains.

In Figure 1 we analyze the behavior of our scheme for a binary memoryless source. We show the probability of block error for biases p ranging from 0.11 to 0.08. Since the code rate is $R = 303/500$, this translates into normalized redundancies $(R - h(p))/h(p)$ ranging from 0.21 to 0.51 where

$$h(p) = p \log_2 \frac{1}{p} + (1-p) \log_2 \frac{1}{1-p}.$$

The coding scheme is based on 8 different realizations of (3,6) regular low-density parity-check matrices with 50 bits allocated to iterative doping, and 3 bits allocated to identifying the LDPC matrix used. Despite the fact that the choice of the code is not "optimized," we observe that the performance is remarkably close to that of nonuniversal arithmetic coding (incorporating knowledge of the source bias). The curve labelled 'Optimal block coding' gives the block-error probability of the (non-constructible) optimal code which assigns a distinct codeword to the most probable 2^{303} source realizations. The curve labelled "information spectrum" (cf. [13]), characterizes the behavior of an 'ideal' variable-length coding scheme where the length of the codeword assigned to $x_1, \ldots x_n$ is equal to $-\log_2 P_{X_1,\ldots X_n}(x_1, \ldots, x_n)$, plotting the probability of the set of source realizations whose 'ideal' codelengths exceed Rn.

Figure 2 gives the results of an analogous experiment with a longer blocklength and a library of 8 rate-$\frac{1}{2}$ irregular LDPCs drawn at random from the ensemble designed for the BSC in [8]. We show the effect of varying the number of bits allocated to doping from 80 to 100. At the point marked at 10^{-6} no errors were detected in 10,000 trials.

Figure 3 shows the results obtained with a 4-state Markov chain with conditional biases equal to $(0.1, 0.6, 0.4, 0.9)$ leading to an entropy rate of 0.5407 bits per symbol. The coding scheme does not code 350 of the source symbols (corresponding to the segments at the output of the BWT with highest entropy) and uses the same scheme used in Figure 2 for the remaining symbols. Varying the number of symbols allocated to doping yields different rate values. Again, the degree of approximation to the ideal coding curve is rather encouraging.

Figure 4 shows the histogram of the output lengths (divided by the blocklength) obtained from 2000 independent trials for another four-state Markov source with entropy rate 0.469 bit/symbol. In this case, we take the longer blocklength $n = 22658$, and we let $n' = 2658$ be the length of the segment of high-entropy symbols that are fed directly to the output. The scheme now operates in variable-length mode with as many doping bits as necessary to achieve perfect decompression, a number which is dictated by the source realization. Also shown in Figure 4 is the histogram of the lengths achieved by an efficient implementation of the Lempel-Ziv algorithm, which suffers a rather severe penalty due to its universality.

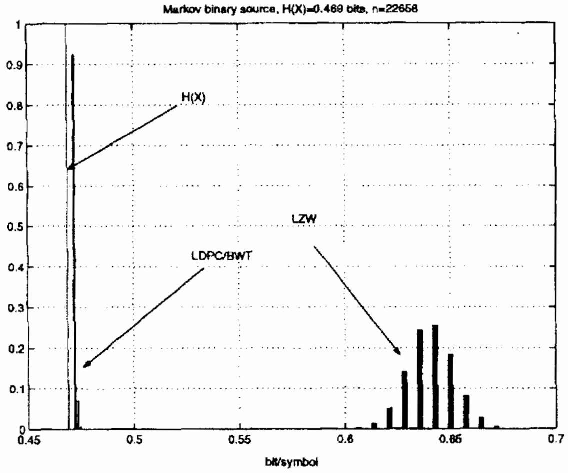

Figure 4: Histogram of compression ratios for a binary four-state Markov chain with entropy 0.469 bits/symbol.

Figure 5 illustrates the very different resilience of arithmetic coding and the new scheme against channel erasures. It should be noted that in neither case we take any countermeasures against the channel erasures. In future work, we report on source/channel schemes derived from the basic scheme of this paper which exhibit much more robustness against channel noise. A source block of 3000 biased coin flips is compressed into a block of 1500 syndrome bits, 200 doping bits and 3 bits identifying the LDPC. The syndrome bits are decompressed by the conventional BP algorithm with the only difference that a subset of the check nodes are now missing due to the channel erasures. As we vary the coin bias from 0.1 to 0.08, we observe an 'error floor' in Figure 5 which is dominated by the fact that the 203 doping and code library bits are sent completely unprotected. The abscissa in Figure 5 is a normalized measure of the distance from the fundamental limit that would be achievable for asymptotically long blocklengths, namely $R \approx \frac{h(p)}{1-e}$ where e is the channel erasure probability.

From the numerical experiments we can draw the conclusion that the new scheme is quite a bit more robust than conventional variable-length schemes against channel erasures, while at the same time performing quite close to the ideal coding performance limits. Naturally, with a more refined channel code selection than the one performed here those limits can be approached even more closely.

REFERENCES

[1] P. E. Allard and A. W. Bridgewater. A source encoding technique using algebraic codes. *Proc. 1972 Canadian Computer Conference*, pages 201–213, June 1972.

[2] T. Ancheta. Syndrome source coding and it universal generalization. *IEEE Trans. Information Theory*, 22, no. 4:432 – 436, July 1976.

[3] J. Bajcsy and P. Mitran. Coding for the Slepian-Wolf problem with Turbo codes. *Proc. 2001 IEEE Globecom*, pages 1400–1404, 2001.

[4] M. Burrows and D. J. Wheeler. A block-sorting lossless data compression algorithm. Tech. Rep. SRC 124, May 1994.

[5] I. Csiszar and J. Korner. *Information Theory: Coding Theorems for Discrete Memoryless Systems*. Academic, New York, 1981.

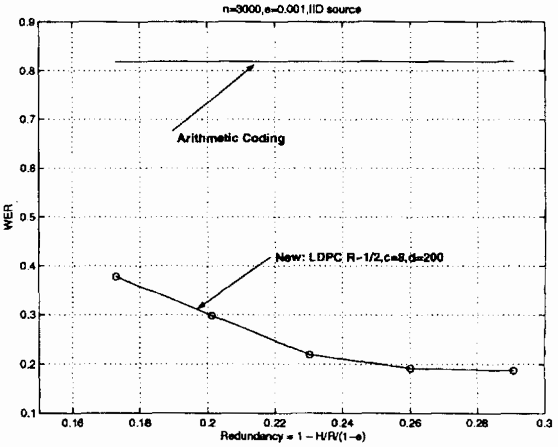

Figure 5: Block error rate of a 3000-to-1703 source code and arithmetic coding for a biased coin a a channel with erasure probability equal to $e = 0.001$

[6] M. Effros, K. Visweswariah, S. Kulkarni, and S. Verdú. Data compression based on the Burrows-Wheeler transform: Analysis and optimality. *IEEE Trans. on Information Theory*, 48:1061–1081, May 2002.

[7] P. Elias. Interval and recency rank source coding: two on-line adaptive variable-length schemes. *IEEE Trans. Inform. Theory*, 33:3–10, Jan. 1987.

[8] R. Urbanke et al. http://lthcwww.epfl.ch/research/ldpcopt/, 2002.

[9] K. C. Fung, S. Tavares, and I. M. Stein. A comparison of data compression schemes using block codes. *Conf. Rec. IEEE Int Electrical and Electronics Conf.*, pages 60–61, Oct. 1973.

[10] R. G. Gallager. *Low-Density Parity-Check Codes*. MIT Press, 1963.

[11] J. Garcia-Frias and Y. Zhao. Compression of correlated binary sources using Turbo codes. *IEEE Communication Letters*, 5:417–419, Oct. 2001.

[12] J. Garcia-Frias and Y. Zhao. Compression of binary memoryless sources using punctured Turbo codes. *IEEE Communication Letters*, 6:394–396, Sep. 2002.

[13] T. S. Han and S. Verdú. Approximation theory of output statistics. *IEEE Trans. Information Theory*, 39:752–772, May 1993.

[14] A. D. Liveris, Z. Xiong, and C. N. Georghiades. Compression of binary sources with side information at the decoder using LDPC codes. *IEEE Communication Letters*, 6, no. 10:440–442, Oct. 2002.

[15] T. Murayama. Statistical mechanics of the data compression theorem. *J. Phys. A: Math. Gen.*, 35:L95L100, 2002.

[16] T. J. Richardson and R. L. Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Trans. Information Theory*, 47:599–618, Feb. 2001.

[17] T. J. Richardson and R. L. Urbanke. Efficient encoding of low-density parity-check codes. *IEEE Trans. Information Theory*, 47:638–656, Feb. 2001.

[18] E. Weiss. Compression and coding. *IRE Transactions on Information Theory*, pages 256–257, Apr. 1962.

[19] A. D. Wyner. Recent results in the Shannon theory. *IEEE Trans. Information Theory*, IT-20, Jan. 1974.