DESIGN AND PERFORMANCE OF A NOVEL LOW-DENSITY PARITY-CHECK CODE FOR DISTRIBUTED VIDEO CODING

João Ascenso[1], Catarina Brites[2], Fernando Pereira[3]

[1]joao.ascenso@lx.it.pt, [2]catarina.brites@lx.it.pt, [3]fp@lx.it.pt
[1]Instituto Superior de Engenharia de Lisboa, [2,3]Instituto Superior Técnico, [1,2,3]Instituto de Telecomunicações

ABSTRACT

Low-density parity-check (LDPC) codes are nowadays one of the hottest topics in coding theory, notably due to their advantages in terms of bit error rate performance and low complexity. In order to exploit the potential of the Wyner-Ziv coding paradigm, practical distributed video coding (DVC) schemes should use powerful error correcting codes with near-capacity performance. In this paper, new ways to design LDPC codes for the DVC paradigm are proposed and studied. The new LDPC solutions rely on merging parity-check nodes, which corresponds to reduce the number of rows in the parity-check matrix. This allows to change gracefully the compression ratio of the source (DCT coefficient bitplane) according to the correlation between the original and the side information. The proposed LDPC codes reach a good performance for a wide range of source correlations and achieve a better RD performance when compared to the popular turbo codes.

Index Terms — Wyner-Ziv video coding, LDPC codes

1. INTRODUCTION

The theoretical foundations of distributed video coding (DVC), the Slepian-Wolf and Wyner-Ziv theorems, suggest that it is possible to independently encode and jointly decode two statistically dependent sources, *X* and *Y*, with the same performance as when the two sources are encoded and decoded together. In DVC, a single decoder performs the joint decoding of all encoded sequences, exploiting the statistical dependencies between them. However, to achieve such target RD performance in a practical DVC system, it is necessary to use powerful channel codes, notably the turbo and low-density parity-check (LDPC) codes.

The main goal of this paper is to design efficient low-density parity-check codes for the distributed video coding (DVC) scenario. The LDPC codes are a class of linear block codes that can approach the Shannon limit quite closely [1] for several types of transmission and storage channels. The most important issue in the design of an LDPC based DVC system is the capability to extract a high number of codes at fine granular compression ratios (or code rates), since in the DVC setting it is necessary to finely adapt the LDPC code compression ratio to the varying statistics of the correlation noise, i.e., to the errors between the side information *Y* and the original data *X*. In a DVC scenario, the correlation noise varies significantly, depending on several factors, such as the motion content of the sequence and the efficiency of the motion estimation/compensation techniques employed at the decoder; therefore, a wide amplitude of compression ratios must be achieved while maintaining a high RD performance. Moreover, the high compression ratio codes must be embedded in the lower compression ratio codes, i.e. the bits received at lower rates must be useful and thus combinable with the additional bits received when a higher rate is further targeted. This type of codes are referred in the literature as rate-compatible codes and are used in solutions (e.g. wireless communications) where the transmission channel statistics vary over time and a feedback channel is available.

Several rate-compatible strategies can be followed to obtain different compression ratios, such as extending, puncturing, or splitting. In [2,3], a practical solution is proposed where a base code is constructed for a high compression ratio (i.e. the minimum amount of bits to send); to obtain lower compression ratios, the base code is recursively split into two smaller codes until the necessary compression ratio is obtained. In [2], extended-Hamming and product-accumulate codes are used as base codes whereas in [3], regular and irregular constructions are considered. In this paper, a different approach is proposed: a low compression ratio (equal to 1:1) LDPC code is first built; then, to obtain higher compression ratios, it is proposed to merge the parity-check nodes of the LDPC base code until the necessary rate is achieved. This novel approach is effective to obtain a wide range of compression rates and allows an effective optimization of the LDPC code structure, a major advantage when compared to previous work [2,3]. The proposed LDPC code structure is optimized for low compression ratios, where certain graph structures (e.g. cycles) can be avoided, since for high compression ratios it is more difficult or even impossible to condition the graph structure, as will be seen in Section 3. The LDPC codes considered here are carefully designed with a selective cycle avoidance algorithm [4] in order to obtain an optimized structure and take into account the node merging operation that will occur to obtain low rate codes.

In the context of DVC, the most common Slepian-Wolf code is still the popular turbo codes. However, the LDPC code proposed in [2] for distributed source coding (DSC) allows higher efficiency and is being applied to DVC codecs, such as the DISCOVER codec [5], a state-of-the-art solution for DVC.

This paper is organized as follows: in Section 2, a brief overview of LDPC codes is presented; in Section 3, the novel techniques to design LDPC codes are proposed and in Section 4 the LDPC code RD performance is evaluated. Finally, in Section 5, some final remarks are drawn.

2. LOW-DENSITY PARITY-CHECK CODES

As illustrated in Figure 1, LDPC codes can be represented graphically via a bipartite graph or factor graph defined by two node types (solid squares and circles in Figure 1): the variable nodes (or v-nodes), which represent the codeword bits, and the check nodes (or c-nodes), which represent the parity-check equations of the code's parity-check matrix *H* with dimension $m \times n$. The matrix *H* is so named because it performs $m = n - k$ parity checks on a received codeword, where *k* represents the amount of data to code and *n* represents the total amount of data (of which $n - k$ is redundant). The parity-check matrix *H* of the LDPC code is sparse, i.e., it has a low density of 1's: $w_r \ll n$ and $w_c \ll m$ with w_r and w_c representing the number of 1's in each row (c-node degree) and column (v-node degree), respectively. The v-nodes are connected by edges to c-nodes according to matrix *H*. Each edge connecting a v-node *j* to a c-node *i* implies $h_{ij}=1$, where variable *j* is a component of the parity-check equation of row h_i. Each of the parity-check equations, when multiplied by a codeword *x*, must fulfill

$Hx = 0$, i.e. the bit values connected to the same c-node must sum to zero. When using LDPC codes for the general case of DSC, the most popular approach is to adopt the scheme suggested by Wyner in 1974 [6], for linear binary block codes. For a certain source x, the encoder calculates the syndrome $s = Hx$ (represented by dotted nodes in Figure 1), and sends it to the decoder; the encoder code rate is, in this case, m/n, i.e., the compression ratio is $n:m$. After, the decoder constructs the side information y, and with the help of a correlation noise model between x and y, it attempts to reconstruct the source x using a belief propagation or maximum likelihood decoding algorithm. This type of approach was first used in [7] for simple codes and extended in [8] to the highly efficient LDPC codes.

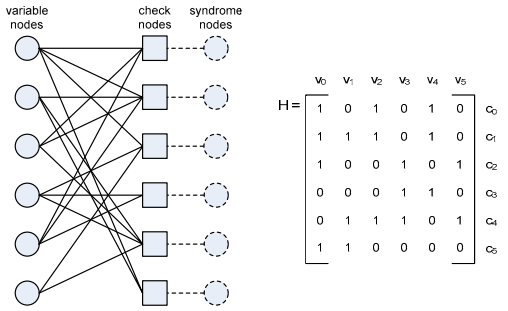

Figure 1 – LDPC syndrome code.

Basically, in the context of LDPC codes, by calculating Hx the encoder maps the n-length input sequence x into one of 2^{n-k} syndromes, through the division of the sequence space (with 2^n sequence possibilities) into 2^{n-k} cosets, each one labeled by one distinct syndrome. All the 2^{n-k} cosets are disjoint and contain 2^k codewords with maximum Hamming distance, which guarantee a good performance over the binary symmetric channel [8].

3. DESIGNING NOVEL LDPC CODES FOR DVC

An LDPC code appropriate for the DVC scenario must fulfill three main requirements: i) *encoding complexity*: must be kept as low as possible in order to allow shifting most of the complexity to the decoder; ii) *rate-compatible strategy*: the code must be rate adaptive and incremental, i.e., the codes at higher rate should be embedded in the codes of lower rates, to allow a dynamic adaptation of the code rate by using a feedback channel; iii) *compression efficiency*: the code must have a high rate performance (ideally close to $H(X|Y)$, the Slepian-Wolf limit) for a wide range of compression ratios, to cope efficiently with changes in the correlation between X and Y. The first requirement is usually met by syndrome based LDPC coding; since the matrix H is sparse, the encoder complexity is kept low and proportional to the number of edges (or 1's) in the LDPC code. Regarding the requirement ii), a novel technique is proposed based on c-node merging to obtain a rate-compatible strategy (Section 3.1), instead of the c-node splitting technique used in [2,3]. The proposed approach has a major advantage: the LDPC high rate base code can be optimized instead of a low rate sub-code, allowing to use powerful graph conditioning techniques (e.g. [4]), which fail when applied to the low rate sub-code, due to their structure (low amount of rows or c-nodes). Thus, to fulfill the requirement iii), the novel strategies to design the LDPC code (Section 3.2) take into account the check node merging technique, make a selective avoidance of cycles in both base and sub-codes and carefully place the variable nodes to break the error bursts typical in the side information.

3.1. Rate-compatible LDPC code by check node merging

This section proposes a technique to obtain a rate-compatible LDPC code by merging any two check nodes, as long as they are connected by a 2-degree syndrome node. When two check nodes are merged, m decreases by one unit, and a higher compression ratio is obtained; the c-nodes merging operation corresponds to the sum of two H rows. The technique to obtain a new (merged) check node, that contains the edges merged of the two old c-nodes, is quite simple, as Figure 2 illustrates:

1. A base code of rate = 1 is generated, corresponding to an H matrix of size 6×6, as shown in the Figure 2 example.
2. Then, c-nodes are merged according to a predefined order: (c_0 and c_1) and (c_4 and c_5), and higher compression ratios (up to 3:2) are successively obtained from the same base code. The lower rate codes obtained are referred here as sub-codes.

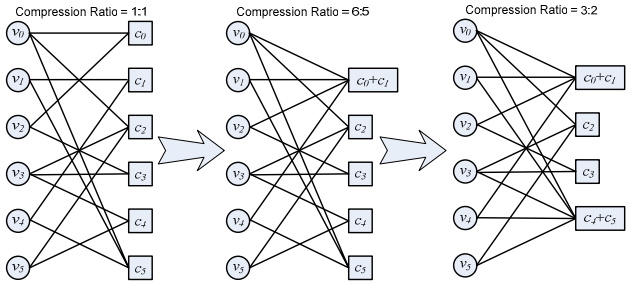

Figure 2 – Factor graphs when check nodes are merged.

This method can provide an elegant way to obtain a rate adaptive LDPC code, since the graph structure can be adapted to obtain fine granular code rates; however, it does not provide a rate-compatible LDPC code in the sense that a set of syndrome bits cannot be combined with previously sent syndrome bits. To obtain this characteristic, it is necessary to include at the encoder, for each pair of c-nodes to merge, a 2-degree syndrome node, as shown in Figure 3.

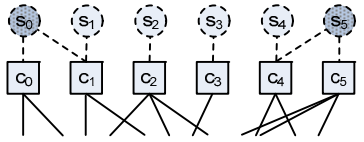

Figure 3 – Syndrome node placement to obtain the graph structure of Figure 2 (shaded nodes are punctured).

To obtain the higher compression ratio 3:2 in Figure 2, the encoder sends all the syndrome bits, except s_0 and s_5 which are punctured. At the decoder, for each syndrome bit not received, the c-nodes connected to it are merged. Considering that 2-degree nodes represent equality, it can be easily proved that the punctured syndromes can be removed as long as their connected c-nodes are merged. At the light of a coset interpretation, this represents the union of two cosets, since there will be $2^{n-(k+1)}$ cosets, each one with 2^{k+1} codewords.

Although this scheme is highly flexible in the choice of the c-nodes to merge, to obtain a good performance for a wide range of code rates, it is necessary to fulfill a certain constraint: the check-node degree distribution of the base code and of the sub-codes must be as concentrated as possible, i.e. all the c-nodes should have no more than two different degrees and their degrees should be as similar as possible. This result is well known in the LDPC channel coding literature [9] and was confirmed in practice for LDPC syndrome codes. A question then arises: Which structure should be used to connect the syndrome nodes to the check nodes? The authors experimented several types of graph structures for the syndrome nodes, and the respective transmission orders (following the above criterion), all of them reaching similar performance. Thus, a simple structure, as shown in Figure 4, where each syndrome node is connected to two adjacent c-nodes, was selected. As it can be noticed, this structure corresponds to the accumulator used in the family of repeat accumulate channel codes [10], which consist of a

concatenation of a set of repetition codes with one or more accumulators and an interleaver; this was also employed in [2,3].

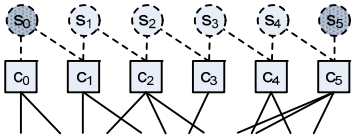

Figure 4 – LDPC syndrome based accumulator.

Once the syndrome nodes placement is defined, it is necessary to define the order by which the check nodes are merged (step 2 above). For the LDPC syndrome based accumulator, the transmission order is defined within a puncturing period Δ, which determines the minimum amount of rate spent in the first transmission, and the granularity of code rates which are incrementally obtained. The proposed algorithm to define the transmission order in Δ for the accumulator structure is described in the following:

1. $c = \Delta$, $l = 1$, send position c in each puncturing period.
2. Send the position $c/2 + i \times c$ with i = {0, …, l-1} in each puncturing period.
3. $l = l \times 2$ and $c = c/2$.
4. If all positions in Δ have been sent, exit; otherwise, go back to 2.

The l, c and i are auxiliary variables to help in the calculation of each position to be sent. This algorithm allows maintaining a concentrated check node degree distribution, each time step 2 is executed.

3.2. Graph conditioning LDPC syndrome code

The performance of the LDPC codes depends on several factors, such as the regularity nature of the graph; irregular LDPC codes can achieve higher bit error rate efficiency [1,8]. An LDPC code is irregular when the degree of the variable and check nodes is not constant across the code. Another important factor is the length of the cycles in the bipartite graph. Since the decoding algorithms, such as the sum-product algorithm (SPA), can achieve optimal decoding only in cycle-free graphs, it is natural to minimize the number of short cycles in the design of the LDPC code. Thus, techniques that limit the effect of cycles in the LDPC code performance, such as the graph conditioning techniques proposed in [4,11], are also necessary when the target is to design efficient LDPC codes for the DVC scenario.

3.2.1. LDPC code features

In order to design an efficient LDPC code when the novel check node merging technique is used to obtain a rate-compatible strategy, it is necessary to take into account the following features:

1. The base code must be designed to maintain a valid structure for any sub-codes, i.e., no more than one edge can connect any c-node/v-node pair. So, to guarantee valid graph structures for the sub-codes, the base code must follow this rule: the c-nodes to merge cannot have a common neighbor, i.e. the base code matrix cannot have ones in the same column in the rows to sum.
2. The LDPC syndrome code can benefit if a graph conditioning technique is applied to the base code. In [4], the concept of stopping steps was proposed: a stopping set is a set of variable nodes which has all its neighbors connected to the set at least twice. The stopping sets impair the code performance when all v-nodes of the set are affected by errors, causing a decoding failure. Thus, it is proposed here to design an LDPC code base matrix (for rate = 1) using the greedy search algorithm ACE (*approximate cycle EMD*) [4] which increases the smallest stopping set size, to obtain better performance for an iteratively decoded LDPC irregular code.
3. Simultaneously, it is also necessary to obtain good sub-codes that result from the parity-check node merging. However, since for high compression ratios the number of c-nodes is quite low and the number of v-nodes and edges remain constant, it is difficult or even impossible to apply the graph conditioning techniques for the sub-codes. Therefore, a simpler option is to forbid certain types of cycles that impair the LDPC code efficiency. So, inspired by the relevant criteria in the literature [9], 4-length cycles that involve only 2-degree v-nodes are forbidden.
4. Another important issue is the proper placement of the v-nodes. As shown in [9,10], low-degree v-nodes are susceptible to errors, because they converge slower than high degree v-nodes and can affect the code efficiency when a significant amount are affected by errors. However, their presence is necessary to have lower degree c-nodes [9]. Considering that the low-degree v-nodes are the most vulnerable ones in the code, they must be placed taking into account the nature of the correlation noise in DVC, where the side information estimation in certain regions fails due to erratic motion, occlusions and/or illumination changes. So, quite often, error bursts in the side information associated to noisy regions are present. To improve the LDPC code capability to correct consecutive bit errors, it is proposed to insert periodically the high degree v-nodes ($w_v > 3$); this avoids that errors bursts only affect low degree v-nodes.

While feature 1 is due to the c-node merging technique, features 2-3 are applied for the first time to improve the performance of a LDPC syndrome code (for high rates), because of the rate-compatible strategy chosen. Finally, feature 4 takes into account the DVC virtual channel statistics to improve the code rate performance.

3.2.2. LDPC code construction algorithm

Considering the above features, and given certain global distributions, $\lambda(x)$ and $\rho(x)$, for the variable and check nodes degrees w_c and w_v, respectively, the algorithm proposed here to create the irregular LDPC base code with $n \times n$ size is the following:

1. Generate variable nodes, one by one, starting from the low-degree v-nodes, according to the $\lambda(x)$ and $\rho(x)$ degree distributions, with random connections (edges) to the c-nodes. The v-node is accepted as valid if:
   a. All v-node edges do not connect more than once to a c-node in the base code and in the lowest rate sub-code. This guarantees that no multi-edges exist, i.e. two or more edges connecting a c-node to a v-node.
   b. The ACE algorithm requirements are met, i.e., all the cycles of length less than a specified threshold (d_{ACE}) have ACE values less than η_{ACE}; refer to [4] for more details.
   c. There are no 4-length cycles involving only 2-degree v-nodes in the lowest rate sub-code.
2. The previous step is repeated until the whole parity check matrix is created.
3. Check if the parity check matrix is full rank (all the rows and columns of H are linearly independent); otherwise, go to step 1. This operation is needed to guarantee that the parity check matrix is invertible in order to recover the original data, independently of the amount of errors in the side information, for rate = 1.
4. In this last step, the placement of v-nodes (feature 4) is done. The algorithm consists in two steps: i) shuffling of all v-nodes with degree $w_v \leq 3$, ii) guarantee that each variable node with degree $w_v > 3$ is equally spaced with a period proportional to the total number of v-nodes with $w_v > 3$.

The major advantage of the proposed LDPC design is that the code graph structure can be tailored for different code rates; in this case, it was optimized for the lowest and highest compression ratios, but different strategies can be applied according to the amount of correlation noise between the side information and the original data.

4. EXPERIMENTAL RESULTS

The proposed LDPC code is here evaluated in the context of a well-

known WZ video codec which follows the Stanford architecture: the DISCOVER codec [5]; since this is one of the state-of-the-art solutions in terms of WZ video codecs, it will be used here for comparison purposes. The two QCIF video sequences considered are: Hall Monitor and Foreman at 15Hz. In all the experiments, only the luminance data is considered for the RD performance evaluation. A GOP length of 2 is used. The key frames are H.264/AVC Intra encoded with quantization parameter values which allow having almost constant decoded video quality for the full set of frames (key frames and WZ frames). The test conditions for the DCT, quantizer, frame interpolation, correlation noise modeling and reconstruction modules are the same as in [5]. Regarding the Slepian-Wolf codec module, three types of channel code solutions are evaluated:

i) Proposed LDPC code with node merging: This code was constructed according to Section 3.2 with $\lambda(x) = 0.131x + 0.26x^2 + 0.187x^6 + 0.115x^7 + 0.08x^{18} + 0.227x^{20}$ and $\rho(x) = 0.17x^3 + 0.83x^4$; these $\lambda(x)$ and $\rho(x)$ distributions obtained from [3]. A rate-compatible strategy was achieved using the technique described in Section 3.1 (node merging). For the ACE algorithm, the pair $(d_{ACE}, \eta_{ACE}) = (13, 7)$ was used.

ii) DISCOVER LDPC codec: A detailed description of the LDPC code used in the DISCOVER codec is presented in [5] and maintains the same edge degree distributions as i). The rate-compatible strategy corresponds to the node splitting technique from [2,3].

iii) DISCOVER Turbo codec: It makes sense to compare the proposed channel code solution with a turbo code based solution since turbo codes also have a bit error rate performance close to the Shannon limit. In this case, the DISCOVER turbo encoder encloses two rate ½ recursive systematic convolutional encoders and an interleaver; the systematic bits produced by the turbo encoder are discarded while the parity bits are stored in a buffer and sent upon decoder request. The turbo decoder is composed by two soft-input soft-output decoders implemented using the logarithmic *maximum a posteriori* (Log-MAP) algorithm.

Note that both i) and ii) LDPC code solutions use the log-domain SPA at the decoder and the inverse of H matrix is used to recover the original data when the compression ratio is 1:1. To detect successful decoding, the parity-check equations must all be satisfied and then an 8-bit Cyclic Redundancy Check (CRC) code is then used to detect residual errors [5].

Table 1 shows for the Hall Monitor and Foreman sequences the WZ rate savings ΔR, in terms of percentage, of the proposed LDPC code with respect to the DISCOVER LDPC codec, $\Delta RLDPC$, and DISCOVER Turbo codec, $\Delta RTurbo$; only the WZ rate is considered since the key frames rate is the same for the three alternative channel code solutions. In Table 1, Q_i represents the i-th WZ quantization matrix associated with the i-th RD point [5]; when Q_i increases, the bitrate and quality also increases. The rate and PSNR columns in Table 1 correspond to the WZ RD performance obtained with the LDPC code proposed in this paper; as expected, the same PSNR values were obtained for the three alternative channel code solutions.

As it can be observed in Table 1, the proposed LDPC code with node merging allows a maximum WZ rate reduction of 19.2% and 10.2% versus the turbo code solution, for the Hall Monitor and Foreman sequences, respectively. When compared with the DISCOVER LDPC codec, the proposed LDPC solution allows WZ rate savings of up to 8% and 2.8%, for the Hall Monitor and Foreman sequences, respectively. For video sequences with higher motion content, like the Soccer sequence, the conclusions are similar to the ones drawn for the Foreman. The higher WZ rate savings are obtained for the Hall Monitor sequence because the proposed LDPC code design has a higher efficiency when the correlation between the side information and the original data is medium or high, i.e. for medium/lower compression ratios. Since the turbo code solution is parity bit based instead of syndrome based, the proposed LDPC code allows a higher WZ rate saving over the turbo code solution, as shown in [12] for the general DSC case. Moreover, when the correlation between the original data and the side information is low, the compression ratio in the turbo code solution can be higher than 1.

Table 1 – WZ rate saving (in %) of the proposed LDPC code regarding DISCOVER Turbo code and DISCOVER LDPC code for the Hall Monitor and Foreman sequences.

Q_i	Hall Monitor			
	Rate [kbps]	PSNR [dB]	$\Delta RTurbo$ [%]	$\Delta RLDPC$ [%]
1	9.77	31.53	17.0	6.5
4	24.53	34.39	19.2	8.0
6	39.10	35.94	18.4	7.5
8	81.79	40.53	14.3	4.4
Q_i	Foreman			
	Rate [kbps]	PSNR [dB]	$\Delta RTurbo$ [%]	$\Delta RLDPC$ [%]
1	22.36	28.35	10.2	2.2
4	60.70	31.90	8.6	2.8
6	93.53	33.16	9.8	2.0
8	211.75	38.65	9.8	1.2

5. FINAL REMARKS

In this paper, a novel LDPC code was designed and evaluated in the context of the DVC scenario. The proposed rate-compatibility strategy makes use of a node merging technique and, according to the experimental results, leads to WZ rate savings up to 8% with respect to the node splitting technique. As future work, it is planned to exploit the memory that exists in the correlation noise between the original data and the side information in the decoding process.

7. REFERENCES

[1] S.-Y. Chung, G. D. Forney, T.J. Richardson, R. Urbanke, "On the Design of Low-Density Parity-Check Codes within 0.0045 dB of the Shannon Limit", *IEEE Commun. Lett.*, Vol. 5, No. 2, pp.58-60, Feb. 2001.

[2] J. Chen, A. Khisti, and D.M. Malioutov, "Distributed Source Coding Using Serially-Concatenated-Accumulate Codes", *IEEE Information Theory Workshop*, San Antonio, TX, USA, Oct. 2004.

[3] D. Varodayan, A. Aaron, and B. Girod, "Rate-Adaptive Distributed Source Coding using Low-Density Parity-Check Codes", *39th Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, USA, Oct./Nov. 2005.

[4] T. Tian, C.R. Jones, J.D. Villasenor, and R. D. Wesel, "Selective Avoidance of Cycles in Irregular LDPC Code Construction", *IEEE Trans. on Communications*, Vol. 52, No. 8, pp. 1242-1247, Aug. 2004.

[5] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, M. Ouaret, "The DISCOVER codec: Architecture, Techniques and Evaluation", *Picture Coding Symposium*, Lisbon, Portugal, Nov. 2007.

[6] A. D. Wyner, "Recent Results in Shannon Theory", *IEEE Trans. Inform. Theory*, Vol. IT-20, pp. 2-10, Jan. 1974.

[7] S. Pradhan and K. Ramchandran, "Distributed Source Coding using Syndromes (DISCUS): Design and Construction", *IEEE DCC*, Snowbird, UT, USA, Mar. 1999.

[8] A.D. Liveris, Z. Xiong and C.N. Georghiades, "Compression of Binary Sources with Side Information at the Decoder using LDPC Codes", *IEEE Commun. Letters*, Vol. 6, No. 10, Oct. 2002.

[9] T. Richardson, M. Shokrollahi, and R. Urbanke, "Design of Capacity-Approaching Irregular Low-Density Parity-Check Codes", *IEEE Trans. Inform. Theory*, Vol. 47, No. 2, pp. 619-637, Feb. 2001.

[10] M. Yang, W.E. Ryan, Y. Li, "Design of Efficiently Encodable Moderate-Length High-Rate Irregular LDPC Codes", *IEEE Trans. on Communications*, Vol. 52, No. 4, pp. 564-571, Apr. 2004.

[11] X.-Y. Hu, E. Eleftheriou, D.-M. Arnold, "Progressive Edge-Growth Tanner Graphs", *IEEE GLOBECOM*, San Antonio, TX, USA, Nov. 2001.

[12] A. Liveris, Z. Xiong, and C. Georghiades, "Distributed Compression of Binary Sources using Conventional Parallel and Serial Concatenated Convolutional Codes", *IEEE DCC*, Snowbird, UT, USA, Mar. 2003.