Slepian-Wolf Coding of Binary Finite Memory Source Using Burrows-Wheeler Transform

Chao Chen, Xiangyang Ji, Qionghai Dai and Xiaodong Liu
Tsinghua University[1]
Beijing, 100084, China

Abstract—In this paper, an asymmetric Slepian-Wolf coding (SWC) scheme for binary finite memory source (FMS) is proposed. The a priori information about the source is extracted from the side information at the decoder by Burrows-Wheeler Transform (BWT). Such information is then utilized for the LDPC code based decoding. Benefiting from the universality of BWT, our coding scheme can be applied to any FMS. Experimental results show that our scheme performs significantly better than the scheme which does not utilize the a priori information for decoding.

Keywords: Slepian-Wolf coding, LDPC code, Burrows-Wheeler Transform, distributed source coding.

1 Introduction

Distributed source coding (DSC) attracts increasing research interests in sensor networks and video coding [1]. Different from conventional joint coding scheme, DSC refers to the coding of two or more dependent sources which cannot access each other. Slepian-Wolf coding (SWC) is the lossless case of DSC when only two sources are involved. Theoretically, SWC can perform as well as conventional joint lossless coding [2]. Nowadays, nearly all the DSC-related applications incorporate SWC codecs.

An important special case of SWC is the asymmetric SWC which losslessly compresses source X with decoder side information Y. With the asymmetric SWC codec, the general SWC can be implemented via the source-splitting scheme proposed in [3]. Hence, the asymmetric SWC codec is the basic component of DSC-related applications. So far, several practical coding schemes for asymmetric SWC have been proposed [4, 5, 6]. Among them, A.Liveris et al.'s LDPC code based scheme presents state-of-the-art coding performance [6]. In this coding scheme, the space of all possible source sequences is partitioned into cosets of a LDPC code. Only the syndrome of the source sequence is transmitted as the coset index. The decoder employs channel decoding algorithm to find the nearest sequence to the side information in the coset which is indexed by the received syndrome. This sequence is then the decoding result.

However, in all existing asymmetric SWC codec designs, it is assumed that the source sequence is i.i.d and equiprobable. When it comes to more complex source statistics, the

[1]This work is supported by the Distinguished Young Scholars of NSFC (Grant No.60525111), and the key project of NSFC (Grant No. 60432030).

encoder should firstly remove the redundancy within the source. However, this increases the complexity of the encoder, and thus is unsuitable for applications such as senor networks or up-link oriented video coding systems which require low-complexity encoder. In this paper, we propose an asymmetric SWC scheme which explores the redundancy of the binary finite memory source (FMS) at the decoder.

It is known that the asymmetric SWC is equivalent to the channel coding problem if the correlation between the source and the decoder side information is treated as a "virtual channel" [7]. Hence, if the source itself is compressible, the asymmetric SWC is equivalent to the source-controlled channel coding problem [8]. Specifically, inspired by the Burrows-Wheeler Transform (BWT)-based source-controlled channel decoding algorithm proposed in [9], we iteratively apply the LDPC decoding and BWT to the side information. The BWT extracts the source statistics from the side information. The statistics is then used as *a priori* information for every LDPC decoding iteration. It is noted that our scheme does not increase the complexity of the encoder. Besides, benefiting from the universality of BWT, our scheme can be applied to any FMS. In this sense, our work is closely related to the problem of Universal Slepian-Wolf Coding (USWC). In [10], Uyematsu investigated this problem and proposed a code construction which could theoretically approach the Slepian-Wolf bound. However, it does not provide the practical encoding and decoding methods. Up to now, USWC is still a very challenging and important open problem.

The rest of this paper is organized as follows. In Section 2, our codec is described. The experimental results are presented in Section 3 followed by the conclusion in Section 4.

2 System description

In this section, the statistical data model is described firstly. Then the background of BWT is briefly introduced, followed by the detailed description of the encoding and decoding schemes.

2.1 Statistical Model

A binary FMS $X = \{x_1, ..., x_n\}$ is defined as follows: Θ is a finite set of states. Each state $\theta_i \in \Theta$ is a finite-length string which consists of several most recent bits of the current bit. No state is a suffix of any other states. Current output bit x_i is generated according to the conditional probability $P(x_i|\theta_i)_{\theta_i \in \Theta}$ defined on current state θ_i. Different from equiprobable i.i.d source, FMS itself is compressible.

Denote the binary side information with $Y = \{y_1, ..., y_n\}$. The correlation between X and Y is modeled as:

$$X \oplus Z = Y, \tag{1}$$

in which Z is an i.i.d binary source and is independent of X. The statistics of X is not known *a priori* at the decoder. According to [11], the lowest bit rate at which X can be losslessly communicated to the decoder is $H(X|Y)$.

2.2 Burrows-Wheeler Transform

Burrows-Wheeler Transform was initially proposed for universal source coding [12]. BWT itself does not compress the source because it just permutes the input sequence into the

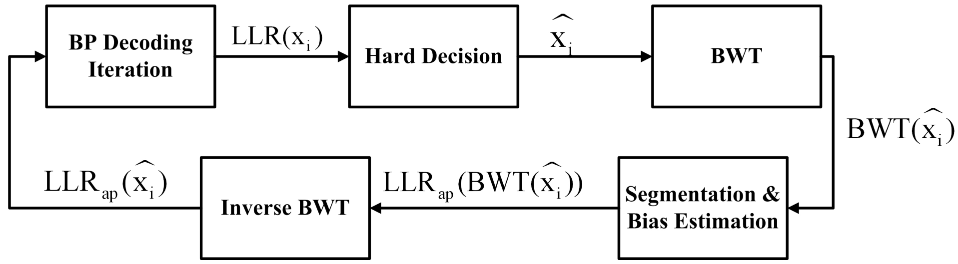

Figure 1: Flowchart of the decoding iteration

output sequence. However, BWT can transform the input FMS into a piecewise i.i.d source [13]. In other words, BWT transforms the redundancy in the memory of FMS into the redundancy in the marginal distributions of the output i.i.d segments. The marginal distribution of every i.i.d segment can be used as *a priori* information for SWC decoding.

2.3 Coding Scheme

At the encoder, the source sequence X is multiplied by the parity-check matrix of a $(n, n-m)$ LDPC code to generate m syndrome bits $S = \{s_1, ..., s_m\}$. These syndrome bits are then transmitted to the decoder. Here, the compression ratio is n/m.

After receiving S, the decoder begins to employ BWT and BP algorithm to iteratively recover X from Y. The flowchart of the decoding iteration is shown in Fig.1. In each iteration, the BP algorithm outputs the *a posteriori* Log-Likelihood Ratio (LLR) of x_i, i.e., $LLR(x_i) \triangleq \log(P(x_i = 0|Y)/P(x_i = 1|Y))$. The hard-decision of LLR is used as the estimation of x_i:

$$\hat{x}_i = \begin{cases} 0 & \text{if} \quad LLR(x_i) \geq 0 \\ 1 & \text{if} \quad LLR(x_i) < 0. \end{cases} \tag{2}$$

Then, the estimation sequence $\hat{x}_i, i \in \{1, ..., n\}$ is permuted by BWT. The input bit $\widehat{x_i}$ is mapped to $BWT(\widehat{x_i})$ by BWT. Here, the output sequence is a piecewise i.i.d sequence. We apply the segmentation algorithm proposed in [14] to adaptively segment the output sequence into several i.i.d segments. If $BWT(\widehat{x_i})$ belongs to the tth segment, we compute the bias $p_t \triangleq P(BWT(\widehat{x_i}) = 1)$ of the segment as:

$$p_t = \frac{n_{t1}}{n_{t1} + n_{t0}}, \tag{3}$$

in which n_{t1} and n_{t0} denote the numbers of occurrences of $1s$ and $0s$ in the tth segment, respectively. The *a priori* LLR is then computed as:

$$\begin{aligned} & LLR_{ap}(BWT(\widehat{x_i})) \\ & = \log \tfrac{P(BWT(\widehat{x_i})=0|y_i)}{P(BWT(\widehat{x_i})=1|y_i)} \\ & = \log \tfrac{P(BWT(\widehat{x_i})=0,y_i)}{P(BWT(\widehat{x_i})=1,y_i)} \\ & = \log \tfrac{P(y_i|BWT(\widehat{x_i})=0)}{P(y_i|BWT(\widehat{x_i})=1)} + \log \tfrac{P(BWT(\widehat{x_i})=0)}{P(BWT(\widehat{x_i})=1)} \\ & = (1 - 2\mathrm{y}_i)\log \tfrac{1-p_z}{p_z} + \log \tfrac{1-p_t}{p_t}, \end{aligned} \tag{4}$$

in which p_z denotes the probability $p_z \triangleq P(z_i = 1)$. The first term derived in (4) represents the *a priori* information about the correlation between X and Y. The second term reflects the *a priori* information about X itself.

After $LLR_{ap}(BWT(\widehat{x}_i))$ is computed, inverse BWT is applied to $LLR_{ap}(BWT(\hat{x}_i))$ to obtain $LLR_{ap}(\hat{x}_i)$. $LLR_{ap}(\hat{x}_i)$ is then incorporated in the next iteration of BP decoding as the *a priori* information. Note that, in the first iteration, $\hat{x}_i$ is initialized as y_i, $i \in \{1, ..., n\}$.

In the following, the modified BP decoding algorithm for LDPC code-based Slepian-Wolf decoding and the segmentation algorithm for the *a priori* information exploration are detailed respectively.

2.3.1 Modified BP decoding algorithm

For the tanner-graph of the LDPC code, $\mathcal{A}_i \subset \{1, ..., m\}$ denotes the set of check-nodes connected with the ith variable-node and $\mathcal{B}_j \subset \{1, ..., n\}$ denotes the set of variable-nodes connected with the jth check-node. In the kth iteration, the message passed from jth check-node to ith variable-node is denoted by $\mu_{j\to i}^{(k)}$. Similarly, $\nu_{i\to j}^{(k)}$ denotes the message passed from ith variable-node to jth check-node.

For the kth BP iteration, $\nu_{i\to j}^{(k)}$ and $\mu_{j\to i}^{(k)}$ are computed as:

$$\begin{aligned} \nu_{i\to j}^{(k)} &= LLR_{ap}(\widehat{x}_i) + \sum_{j' \in \mathcal{A}_i - \{j\}} \mu_{j'\to i}^{(k-1)}, \\ \mu_{j\to i}^{(k)} &= (1 - 2s_j)\tanh^{-1}\Big(\prod_{i' \in \mathcal{B}_j - \{i\}} \tanh\big(\frac{\nu_{i'\to j}^{(k)}}{2}\big)\Big), \end{aligned} \tag{5}$$

where s_j is the jth syndrome bit. In the first iteration, we set $\mu_{j\to i}^{(0)} = 0$ for all $j \in \{1, ..., m\}$. In the end of every iteration, the *a posteriori* information $LLR(x_i)$ is then computed as:

$$LLR(x_i) = LLR_{ap}(\widehat{x}_i) + \sum_{j \in \mathcal{A}_i} \mu_{j\to i}^{(k)}. \tag{6}$$

Then $LLR(x_i)$ is substituted in (2) for the next iteration.

2.3.2 Segmentation method

In our codec, the output of BWT is a piecewise i.i.d sequence. An adaptive segmentation algorithm inspired by [14] is employed to find the transition points between the successive i.i.d segments.

In [14], two segmentation algorithms are proposed. One is a sequential algorithm and the other is non-sequential. In this work, the non-sequential algorithm is employed. This algorithm uses a two-pass scheme to estimate the transition points. For each passes, the BWT output is firstly uniformly partitioned into blocks. Then a metric is utilized to measure the local statistics. If a block is neither the first nor the last block, it is called an inner block. The metric of the kth inner block is defined as follows:

$$M(k) \triangleq H(k-1, k+1) - \frac{1}{2}H(k-1) - \frac{1}{2}H(k+1), \tag{7}$$

where $H(k-1)$ and $H(k+1)$ denote the empirical entropy of blocks $k-1$ and $k+1$, respectively. $H(k-1, k+1)$ is the empirical entropy computed on the concatenation of blocks $k-1$ and $k+1$.

For simplicity, the blocks $k-n, ..., k-1, k+1, ..., k+n$ are termed as the n-block neighborhood of the kth block. In the first pass, the input sequence (i.e. the output of BWT) is uniformly partitioned into blocks of length k_1. Then the metric is computed for each inner block. We select the block with the largest metric value and eliminate all the blocks in its 2-block-neighborhood. For the blocks which were neither selected nor eliminated, we select the block with the largest metric value and, again, eliminate all its 2-block-neighbors. This procedure repeats until all the blocks are either selected or eliminated.

In the second pass, for each of the selected block, the 1-block-neighbors and the selected block itself is uniformly partitioned into $3k_1/k_0$ sub-blocks. The length of each sub-block is k_0. Then the metric is computed for each sub-block. The center of the sub-block which gives the largest metric value is estimated as the transition points.

The block lengths k_1 and k_0 in both passes are chosen as [15, Proposition 2]:

$$\begin{aligned} k_1 &= (\log_2 N)^{1+\alpha} \\ k_0 &= (\log_2 \log_2 N)^{1+\beta}, \end{aligned} \tag{8}$$

where $\alpha, \beta > 0$ are parameters between 2 and 3. k_1 should be divisor of the source length n. k_0 should be divisor of $3k_1$.

3 Experimental results and discussions

3.1 Performance Evaluation

To test the coding performance of our codec, a 3-state tree-source is used as the FMS sequence X. The state of x_i is uniquely determined by its most recent bits. As shown in Fig.2, the state set Θ is $\{1, 10, 00\}$. At state $1, 10, 00$, the bias of x_i is $0.1, 0.3$ and 0.5, respectively. The side information Y is obtained by adding an i.i.d binary source Z with

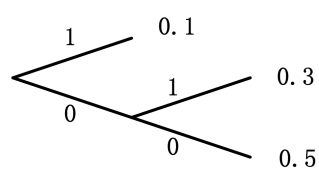

Figure 2: 3-state tree-source

bias p_z to X. By adjusting the bias value p_z, different conditional entropy $H(X|Y)$ value can be obtained. The method for conditional entropy calculation is detailed in the Appendix Section. It should be noted that $H(X|Y) \neq H(p_z)$.

According to the correlation model in equation (1), X and Y is actually correlated via a "virtual" BSC channel. A (10000,5000) LDPC code optimized on BSC channel is employed for coding [14, example 2]. In every iteration, we employ segmentation algorithm proposed in [14] to obtain the i.i.d segments. The level 1 and level 0 block lengths are set as 500 and 50, respectively. The maximum iteration number for BP decoding is 100. The simulation continues until 30 frame errors are collected. The coding performance is presented in Fig.3. At the $H(X|Y)$ of 0.395 bits/sample (bps), BER of 10^{-5} is achieved.

For comparison, we have tested the performance of a "blind" decoding scheme which does not utilized the *a priori* information. Specifically, the second term of (4) is omitted.

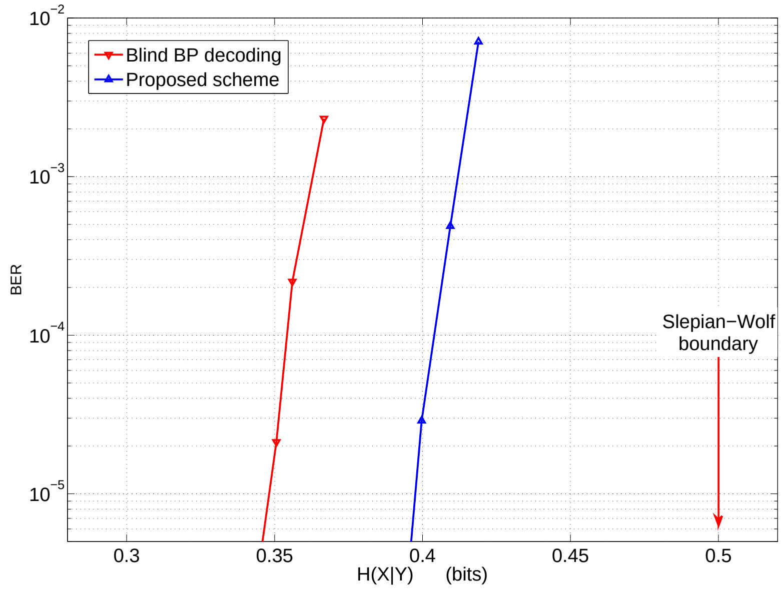

Figure 3: Coding performance

As shown in the figure, our proposed scheme performs significantly better than the "blind" BP decoding scheme.

To help the readers easily understand the inner mechanism of the codec, we present the statistics of $BWT(\widehat{X})$ for a realization of X in Fig.4. We set p_z as 0.095. With our codec, the source is successfully decoded after 16 iterations of decoding. The estimated bias parameters, i.e. p_t in equation (3), is plotted in the figure. According to the structure of source X, the stationary distribution for state "00", "10" and "1" are 0.3987, 0.2848 and 0.3165, respectively (see Appendix). The corresponding bias parameters are 0.5, 0.3 and 0.1, respectively. These true parameters are plotted in red line in the figure. It is observed that, as while as the decoding proceeds, the estimated bias parameter of $BWT(\widehat{X})$ quickly converges to the true parameters.

3.2 Discussions

3.2.1 Universality

It is noted that in the decoding process, the statistics of X is not known *a priori*. The state set Θ and the bias parameters of each state are not explicitly transmitted to the decoder. In this sense, our scheme is universal with respect to the statistics of X.

In the mean while, it is also noted that the bias of Z should be known *a priori* at the decoder to compute LLR_{ap} in equation (4). In this sense, our scheme is not universal with respect to the correlation between X and Y.

Without sacrificing the universality respect to X, we can shift the BWT from the decoder to the encoder. The bias parameters of X can be directly computed at the encoder and sent to the decoder. The bit rate for signaling these parameters is insignificant. Because the parameters are directly computed from X, they are more accurate than those computed at

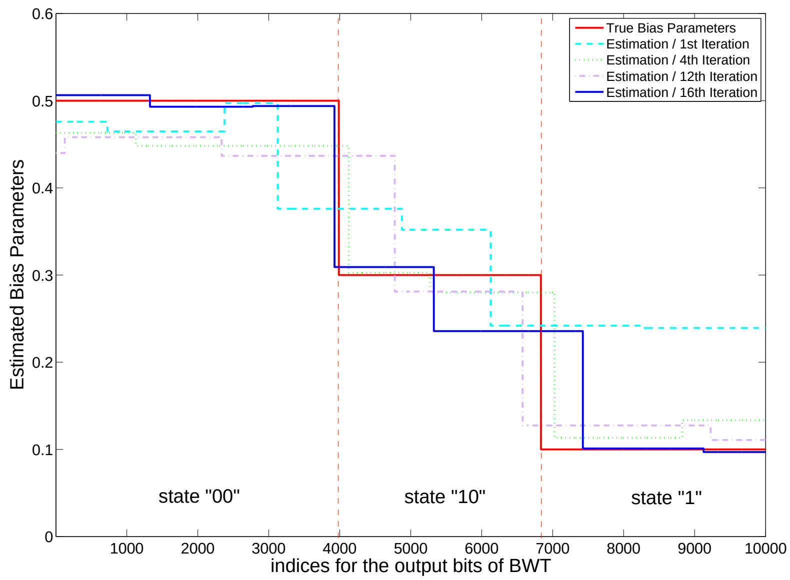

Figure 4: Estimated bias parameters of $BWT(\widehat{X})$ in each iteration

the decoder, and thus, the decoding process will be accelerated. Apparently, it is noted that this scheme increases the complexity of the encoder.

3.2.2 Performance gap

When the states of the tree source are heavily biased, the input distribution of the "virtual channel" will deviate from the capacity approaching distribution. That will give rise to some performance loss. In Table.1, the maximum achievable conditional entropy $H(X|Y)$ at BER of 10^{-5} is listed for different bias parameters of state "1". It is observed that the gap between $H(X|Y)$ and the code rate (i.e. 0.5bps) becomes larger as the bias parameter decreases. Hence, designing more sophisticated Slepian-Wolf codes to alleviate the performance loss is a challenging future work.

Table 1: Performance gap to Slepian-Wolf bound

Bias for state "1"	$H(X\|Y)$ (bps)	Gap to bound (bps)
0.1	0.395	0.105
0.2	0.411	0.089
0.4	0.416	0.084

4 Conclusion

In this paper, we address the asymmetric Slepian-Wolf coding problem for binary finite memory sources. The source redundancy due to the memory is exploited via iteratively applying BWT and BP algorithm to the side information. No information about the source statistics is required *a priori* at the decoder. The decoding algorithm can be universally applied to any FMS. Experimental results show that our scheme can closely approach the theoretical bound when sources are not heavily biased.

5 Acknowledgement

Thanks to Prof. Z.Xiong for helpful discussion and Dr. D.Baron for his comments on the draft.

Appendix: The Calculation of Conditional Entropy

The conditional entropy rate of the FMS source employed in our experiment can be calculated as follows: Firstly, stationary distribution of each state is calculated as p_1, p_{10}, p_{00}. Then we calculate the conditional entropy on each state as h_1, h_{10} and h_{00} using p_z and the bias parameters of each state. The overall conditional entropy rate is calculated as

$$H(X|Y) = p_1 \times h_1 + p_{10} \times h_{10} + p_{00} \times h_{00}. \tag{9}$$

For example, as for the source in Fig.2, the stationary distribution of each state can be obtained by solving equation:

$$\begin{bmatrix} p_1 & p_{10} & p_{00} \end{bmatrix} \begin{bmatrix} 0.1 & 0.9 & 0 \\ 0.3 & 0 & 0.7 \\ 0.5 & 0 & 0.1 \end{bmatrix} = \begin{bmatrix} p_1 & p_{10} & p_{00} \end{bmatrix},$$

in which the square matrix is the probability transition matrix as defined in [17].

Then we calculate the conditional entropy h_1, h_{10} and h_{00}. It should be noted that the conditional entropy is not $h(p_z)$ because Y is dependent on Z. We take the calculation of h_1 as an example. According the definition of conditional entropy, we have

$$h_1 = H(X|Y) = P(Y=0)H(X|Y=0) + P(Y=1)H(X|Y=1), \tag{10}$$

in which $P(Y=0) = (1-p_z)(1-p_x) + p_z p_x$ and $P(Y=1) = 1 - P(Y=0)$. Here, p_x is the bias parameter of state "1". To calculate $H(X|Y=0)$ and $H(X|Y=1)$, the conditional probability $P(X=1|Y=0)$, $P(X|Y=1)$, $P(X=0|Y=0)$ and $P(X=0|Y=0)$ are firstly calculated as follows:

$$
\begin{array}{l}
P(X=1|Y=0) \\
= \frac{P(X=1,Y=0)}{P(Y=0)} = \frac{P(Y=0|X=1)P(X=1)}{P(Y=0)} = \frac{P(Z=1)(X=1)}{P(Y=0)} \\
= \frac{p_z p_x}{(1-p_z)(1-p_x)+p_z p_x} \\
P(X=1|Y=1) \\
= \frac{P(X=1,Y=1)}{P(Y=1)} = \frac{P(Y=1|X=1)P(X=1)}{P(Y=1)} = \frac{P(Z=0)(X=1)}{P(Y=1)} \\
= \frac{(1-p_z)p_x}{(1-p_z)p_x+(1-p_x)p_z} \\
P(X=0|Y=0) = 1 - P(X=1|Y=0) \\
P(X=0|Y=1) = 1 - P(X=1|Y=1)
\end{array}
\tag{11}
$$

Then, $H(X|Y=0)$ and $H(X|Y=1)$ can be calculated as:

$$
\begin{array}{l}
H(X|Y=0) \\
= -P(X=1|Y=0)\log_2 P(X=1|Y=0) - P(X=0|Y=0)\log_2 P(X=0|Y=0) \\
H(X|Y=1) \\
= -P(X=1|Y=1)\log_2 P(X=1|Y=1) - P(X=0|Y=1)\log_2 P(X=0|Y=1)
\end{array}
\tag{12}
$$

Substituting the results of (12) into equation (10), h_1 can be derived. h_{10} and h_{00} can be derived in the same manner. The conditional entropy rate is finally calculated with equation (9).

References

[1] Z. Xiong and A. Liveris and S. Cheng, "Distributed source coding for sensor networks," *IEEE Signal Process Mag.*, vol.21 pp. 80-94, 2004.

[2] J. D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inf. Theory*, vol.IT-19 pp. 471-480, 1973.

[3] B.Rimoldi and R. Urbanke, "Asynchronous Slepian-Wolf coding via source-splitting," in *Proc. IEEE Int. Symp. Information Theory,* Ulm, Germany, Jun. 1997, p.271.

[4] S.Pradhan and K.Ramchandran, "Distributed source coding using syndromes (DISCUS): Design and construction," *IEEE Trans. Inf. Theory*, vol.IT-49 pp. 626-643, 2003.

[5] A.Aaron and B.Girod, "Compression with side information using turbo codes," in *Proceedings of 2002 Data Compression Conference*, pp. 252-261, 2002.

[6] A.D.Liveris and Zixiang Xiong and C.N.Georghiades, "Compression of binary sources with side information at the decoder using LDPC codes," *Communications Letters, IEEE*, vol.6, pp. 440 - 442, 2002.

[7] A.Wyner,"Recent results in the Shannon theory," *IEEE Trans. Inf. Theory*, vol.IT-20 pp. 2-10, 1974.

[8] J. Hagenauer, "Source-controlled channel decoding",*IEEE Trans. Inf. Theory*, vol.IT-43 pp. 2449-2457, 1995.

[9] L. Wang and G. I. Shamir, “BWT based universal lossless source constrolled channel decoding with low density parity check codes”, in *Proceedings of 2005 Data Compresion Conference*, pp. 487, 2005.

[10] T. Uyematsu, “Universal coding for correlated sources with memory, in Canadian Workshop Inform. Theory, Vancouver, June 2001.

[11] T. M. Cover, “A proof of the data compression theorem of Slepian andWolf for ergodic sources, IEEE Trans. Inform. Theory, vol. 21, no. 2, pp. 226-228, Mar. 1975.

[12] M. Burrows and D.J. Wheeler,“A block-sorting lossless data compression algorithm,” Digital System Research Center, Palo Alto, CA, May 10, 1994.

[13] M. Effros, K. Visweswariah, S.R. Kulkarni and S. Verdu, “Universal lossless source coding with the Burrows Wheeler Transform,” *IEEE Trans. Inf. Theory*, vol.IT-48, pp. 1061-1081, 2002.

[14] G.I. Shamir, D.J.Costello, Jr., “Asymptotically optimal low complexity sequential lossless coding for piecewise stationary memoryless sources - Part I: The regular case,” *IEEE Trans. Inf. Theory*, vol.IT-46, pp. 2444-2467, 2000.

[15] S.-Y. Chung, “On the construction of some capacity-approaching coding schemes,” Ph.D. dissertation, Massachusetts Institute of Technology, 2000.

[16] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, “Design of capacity-approaching irregular low-density parity-check codes,” *IEEE Trans. Inform. Theory*, vol. IT-47, pp. 619-637, 2001.

[17] T.M.Cover and J.A.Thomas, ”Elements of Information Theory”, John Wiley & Sons, Inc., Hoboken, New Jersey, 2006, pp. 72.