www.ietdl.org

Published in IET Communications
Received on 9th November 2011
Revised on 10th January 2012
doi: 10.1049/iet-com.2011.0840

ISSN 1751-8628

Efficient joint source–channel decoding of multi-state Markov sequences

H. Kim[1] *D. Har*[1] *Z.-H. Mao*[2] *M. Sun*[3] *H.-N. Lee*[1]

[1]*School of Information and Mechatronics, Gwangju Institute of Science and Technology (GIST), Gwangju, Republic of Korea*
[2]*Department of Electrical and Computer Engineering, University of Pittsburgh, PA, USA*
[3]*Department of Neurological Surgery, Bioengineering and Electrical Engineering, University of Pittsburgh, PA, USA*
E-mail: heungno@gist.ac.kr; hardon@gist.ac.kr

Abstract: In this study, joint source–channel decoding for non-binary source samples is conducted. The non-binary source samples can be modelled as the output of a multi-state Markov chain (MC). As the source samples are directly transmitted after channel coding without source compression, the transmitted signals can be highly correlated. At the receiver, the multi-state MC module can be designed to exploit the statistical correlation of source samples to improve the error correcting performance. However, as the number of states is increased, the multi-state MC module requires high computational complexity. To alleviate this problem, a simplified MC module is proposed. In the simplified MC module, the multi-state MC is replaced with multiple number of two-state MCs each of which exploits bit-level correlation of samples. Simulation results demonstrate that the simplified MC module can lead to competitive reduction in the required signal-to-noise ratio in comparison with the multi-state MC module with reduced computational complexity.

1 Introduction

Shannon's source–channel separation principle is that there is a separable source and channel coding scheme that allows transmission over the channel with arbitrarily low error probability [1]. Under this principle, source coding and channel coding can be independently established for stationary sources and channels [2]. However, considering system resources in terms of bandwidth, delay and complexity, it is sometimes too expensive to implement the separable source and channel coding to achieve optimal performance. For limited resource transmission systems such as wireless sensor devices, for example, joint source–channel coding schemes have received considerable attention as a good alternative to the source–channel separation principle and enhance the performance of coding systems (see [3, 4]).

There are various joint source–channel coding schemes: some includes both source coding and channel coding in a single system or others employ only one of them. One of the most popular schemes in the former is the unequal error protection scheme in which different grades of importance are used for source bits and for providing unequal level of protection against random channel errors. More important source bits that cause higher distortion when damaged, for example, get higher level of error protection [5]. For the latter, there are some source coding schemes that provide channel error protection as well, without using an explicit channel coding scheme. For example, residual redundancy that may be caused by imperfection of a practical source encoder is utilised by the source decoder to enhance the quality of reconstructed signal [6]. In [7–9], the so-called 'error concealment' techniques applied at the receiver exploit residual redundancy in spatial and temporal domain of video and image signals. In some approaches, uncompressed quantised samples are transmitted after channel coding, that is, without any explicit source coding. In [10], an iterative maximum a posteriori (MAP) joint source–channel decoding scheme is proposed to decode the uncompressed source sample sequence. The redundancy between source samples is modelled as a hidden Markov chain (MC) and exploited within the process of the MAP decoder.

The joint source–channel decoder in [10], however, is only for binary source samples. In many practical applications, such as in sensor networks and video/image applications, non-binary multilevel source samples are obtained [11–14]. The multilevel source samples can be modelled as the output of a multi-state MC [13, 14]. The decoding process in [10] thus can be combined with the multi-state MC module; but this scheme requires enormous computational complexity as the number of states is increased. The complexity of the multi-state MC module is $O(KM^2)$, where K is the number of samples and M is the number of states, that is, the alphabet size of a sample [15].

In this paper, research on joint source–channel decoding for non-binary source samples is conducted. To reduce the computational complexity of the multi-state MC module for correlated samples, a simplified MC module is proposed. As the sequence of non-binary samples is converted into

the corresponding binary sequence, we note there exists bit-level correlation that is exploited in this paper. Therefore a multi-state MC can be replaced with multiple two-state MCs which alleviates the computational load for estimating source correlation at the receiver. In this paper, the multi-state MC and the multiple two-state MCs are, respectively, called the sample-level Markov model (SMM) and the bit-level Markov model (BMM). With BMM, we can obtain competitive signal-to-noise ratio (SNR) gain in comparison with SMM.

Section 2 discusses the system model including the multi-state MC which is defined as a source model to produce correlated non-binary source samples. In Section 3, joint source–channel decoding scheme is explained to decode correlated source samples. In Section 4, BMM is proposed and its transition probabilities are provided. To justify the usefulness of BMM, entropy rate is introduced as a criterion for closeness between the two models. This analysis on entropy rates would give useful insights without an extensive computer simulation-based comparison which is time consuming. Section 5 shows the entropy rates and bit-error-rate (BER) performance of BMM and SMM. Finally, conclusions are drawn in Section 6.

2 System model

2.1 Source model

The non-binary source sample, denoted as x_k, where k is the time index, is generated by a stationary first-order MC with transition probabilities $a_{ij} = \Pr[x_k = j | x_{k-1} = i]$ for $0 \le i, j \le M-1$. At first, the sequence of source samples is divided into the blocks of length K which is denoted by $\boldsymbol{x} = [x_1, \ldots, x_K]^{\mathrm{T}}$. As shown in Fig. 1, each block of samples $\boldsymbol{x}$ is converted to a block of bits $\boldsymbol{b}$ of length N_m. The bit sequence $\boldsymbol{b}$ is the message vector which is the input to a binary low-density parity-check (LDPC) encoder [16]. To simplify the notation, the alphabet size M of MC is assumed to be a power of two, that is, $M = 2^L$, where L is the length of binary bits of a source sample, so that $N_m = KL$. Each source sample x_k of the base-10 integer numeral is converted to binary bits of the base-2 equivalent, that is, $\boldsymbol{b} = [b_{1,1}, \ldots, b_{1,L}, \ldots, b_{K,1}, \ldots, b_{K,L}]^{\mathrm{T}}$, where $b_{k,l}$ indicates the lth bit of the kth sample.

2.2 Encoding and modulation

The output of the LDPC encoder is the coded sequence $\boldsymbol{c} = [c_1, c_2, \ldots, c_N]^{\mathrm{T}}$, where N is the length of the coded sequence. The LDPC code with coding rate N_m/N is a linear block code which is specified by an $(N - N_m) \times N$ sparse parity-check matrix $\boldsymbol{H}$. The parity-check matrix can be represented by a bipartite graph, which consists of N variable nodes representing the coded sequence and $N - N_m$ check nodes representing the parity-check equations. An edge in the graph is placed between variable node i and check node j if $H_{ji} = 1$. Each check node is connected to coded sequence whose sum modulo-2 should be zero, that is, $\boldsymbol{Hc} = 0$. Irregular LDPC codes are characterised by two polynomials $\lambda(\alpha) = \sum_{i=1}^{d_{\mathrm{v}}} \lambda_i \alpha^{i-1}$ and $\rho(\alpha) = \sum_{i=1}^{d_{\mathrm{c}}} \rho_i \alpha^{i-1}$, where λ_i is the fraction of the edges in the bipartite graph that are connected to the variable nodes of degree i, ρ_i is the fraction of edges that are connected to the check nodes of degree i, d_{v} is the maximum variable node degree and d_{c} is the maximum check node degree. Regular LDPC codes are specified by the pair $(d_{\mathrm{v}}, d_{\mathrm{c}})$, that is, $\lambda(\alpha) = \alpha^{d_{\mathrm{v}}-1}$ and $\rho(\alpha) = \alpha^{d_{\mathrm{c}}-1}$.

The coded sequence $\boldsymbol{c}$ is modulated via binary phase-shift keying (BPSK) into a bipolar sequence $\boldsymbol{d} = [d_1, d_2, \ldots, d_N]^{\mathrm{T}}$ with $d_i = 2c_i - 1$ and then transmitted over the additive white Gaussian noise channel. The received symbol vector is denoted by $\boldsymbol{y} = [y_1, y_2, \ldots, y_N]^{\mathrm{T}}$ which is given by

$$\boldsymbol{y} = \sqrt{\frac{2E_{\mathrm{s}}}{N_{\mathrm{o}}}}\boldsymbol{d} + \boldsymbol{z} \tag{1}$$

where $\boldsymbol{z}$ is the independent identically distributed Gaussian noise vector with zero mean and the unit variance, N_{o} is the single-sided noise power spectral density and E_{s} is the energy of the BPSK symbol. In (1), $E_{\mathrm{s}}/N_{\mathrm{o}}$ is the SNR under the assumption that symbol rate is equal to the bandwidth of the channel.

To recover the transmitted sequence of source samples, which is equivalent of reconstructing the transmitted bit sequence, the receiver utilises a joint decoding scheme that includes the LDPC decoder and the MC module. The LDPC decoding module takes the received symbol vector $\boldsymbol{y}$ and cooperates with the MC module via turbo-iteration, and produce a final decoded bit sequence $\hat{\boldsymbol{b}}$. The decoding process is explained in the next section.

3 Joint source–channel decoding scheme

In this paper, the M-state MC module is combined with the LDPC decoding module, whereas the two-state MC module is connected to a channel decoding module for turbo codes in [10]. The LDPC decoding module generates the maximum posterior probability on each bit in the coded sequence by the standard message passing algorithm [16]. The MC module produces prior probability for the bit sequence $\boldsymbol{b}$ by the forward–backward algorithm [17]. The two modules cooperate with each other by exchanging the so-called 'extrinsic' information [18]. This exchange of extrinsic information between the two modules in one complete round is referred to as 'super-iteration' in this paper. By the super-iteration, performance in BER of the joint source–channel decoding scheme is improved.

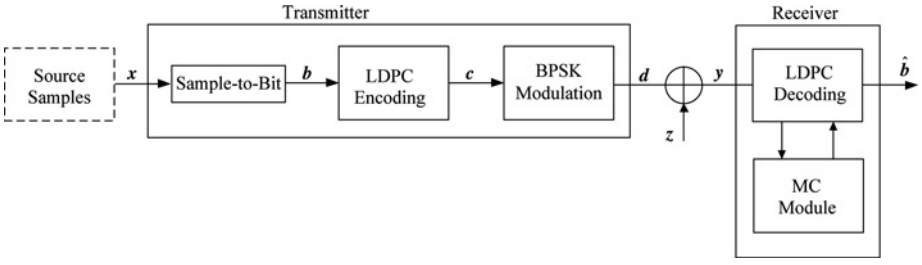

Fig. 1 *Block diagram of the transceiver*

3.1 LDPC decoding module

The message passing algorithm that is applied to the LDPC decoding module is well known. The process is summarised in this section for a matter of completeness but more details can be found in [16]. The description should be compact but complete so that reproduction of the results in this paper can be made without algorithmic ambiguity. Note that we also explain the process for conversion of the output of the LDPC decoding module to make connection with the M-state MC module.

The decoder takes the prior probability on a bit value, that is, 1 or 0, from both the channel decoding and the MC module, respectively. We use log likelihood ratio (LLR) to represent the extrinsic information, since it is well known to be more efficient than exchanging the probabilities. The prior LLRs, denoted as $R_{C,n}$'s, are described as follows

$$R_{C,n} = \begin{cases} R'_{C,n} + O_{\mathrm{SE},n}, & 1 \le n \le N_m \\ R'_{C,n}, & N_m + 1 \le n \le N \end{cases} \tag{2}$$

where $R'_{C,n}$'s and $O_{\mathrm{SE},n}$'s are, respectively, the LLRs from the channel output $\boldsymbol{y}$ and from the MC module. The prior LLRs are given by $R'_{C,n} = (4E_s/N_o)y_n$, and stay the same throughout the whole decoding process for a particular coded sequence. The prior LLRs from the MC module, $O_{\mathrm{CE},n}$'s, are the extrinsic LLRs of the MC module, and get updated from the extrinsic LLRs of the decoder, denoted as $O_{\mathrm{SE},n}$'s, at each super-iteration. At the start of the super-iteration, we set $R_{C,n} = R'_{C,n}$ for all n, since the extrinsic LLRs $O_{\mathrm{SE},n}$'s of the MC module are not available.

The posterior LLRs, denoted as $O_{C,n}$'s, of the decoder are generated by enforcing the parity-check relations, which can be represented as

$$O_{C,n} = R_{C,n} + \ln \frac{\Pr[S|c_n = 1, \boldsymbol{y}]}{\Pr[S|c_n = 0, \boldsymbol{y}]} \tag{3}$$

where S is the event that all the parity-check equations participate on bit n are satisfied simultaneously [16].

Decoder takes the previous posterior LLRs as another prior LLRs and applies the parity-check relationship again, and this process is repeated until the zero syndrome vector is obtained or the number of iterations is less than a prescribed number. The final decoded bit sequence $\hat{\boldsymbol{b}}$ is obtained from the posterior LLRs, $O_{C,n}$'s, for the message bits at the last iteration. From the posterior LLRs, $O_{C,n}$'s, the extrinsic LLRs $O_{\mathrm{CE},n}$'s of the decoder are used as the input to the MC module, which is given by

$$O_{\mathrm{CE},n} = O_{C,n} - O_{\mathrm{SE},n} \tag{4}$$

In (2), the input to the LDPC decoding module includes the LLRs from MC module, $O_{\mathrm{SE},n}$'s. By subtracting $O_{\mathrm{SE},n}$'s from $O_{C,n}$'s, $O_{\mathrm{CE},n}$'s contain the new information on the message bits obtained only from the decoder, which is extrinsic to the MC module.

To use extrinsic LLRs, $O_{\mathrm{CE},n}$'s, in the MC module, we need to transform them to the bit-to-sample probability, which will be denoted by $g_k(j)$'s for $0 \le j \le M-1$. It is because the MC module is employed in a sample-level trellis as shown in Fig. 2. In other words, a string of L message bits, $u_1(j)u_2(j), \ldots, u_L(j)$, composes a single sample j, that is

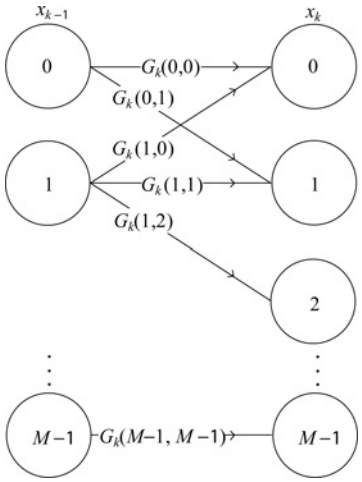

Fig. 2 *Trellis diagram of the M-state MC*

$$j = \sum_{l=1}^{L} u_l(j) 2^{L-l} \tag{5}$$

where $u_l(j) \in \{0, 1\}$ and is the lth bit of the string. Since $O_{\mathrm{CE},n} = \ln(\Pr[b_{k,l} = 1]/\Pr[b_{k,l} = 0])$, each bit probability by the decoder is obtained as

$$\begin{cases} \Pr[b_{k,l} = 1] = \dfrac{\exp(O_{\mathrm{CE},n})}{1 + \exp(O_{\mathrm{CE},n})} \\ \Pr[b_{k,l} = 0] = \dfrac{1}{1 + \exp(O_{\mathrm{CE},n})} \end{cases} \tag{6}$$

where $\Sigma_i \Pr[b_{k,l} = i] = 1$ and $n = (k-1)L + l$. Then, the bit-to-sample probabilities $g_k(j)$'s are converted from the bit probabilities that are given by

$$g_k(j) = \prod_{l=1}^{L} \Pr[b_{k,l} = u_l(j)] \tag{7}$$

where the message bits are assumed to be independent with each other.

3.2 MC module

To exploit the statistical properties of received samples in the MC module, the famous forward–backward algorithm [17] is used. We briefly summarise forward–backward algorithm using the notations of this paper.

It is mentioned that the MC module provides the extrinsic LLRs $O_{\mathrm{SE},n}$'s for the prior LLRs on the message bits as in (2). The MC module is constructed by the sample-transition gains which is denoted by $G_k(i, j)$'s. The kth trellis section of the MC in Fig. 2 indicates the sample-transition gains $G_k(i, j)$'s. The sample-transition gains $G_k(i, j)$'s are the weighted transition probability for the edge from sample i to sample j at the kth trellis which is specified as

$$G_k(i, j) = a_{ij} g_k(j) \tag{8}$$

for $0 \le i, j \le M-1$. With the sample-transition gains $G_k(i, j)$'s, the jth sample probabilities, $C_k(j)$'s, are obtained

doi: 10.1049/iet-com.2011.0840

by forward–backward algorithm [17] as follows

$$C_k(j) = \frac{\sum_{i=0}^{M-1} A_{k-1}(i) G_k(i, j) B_k(j)}{\sum_{i=0}^{M-1} \sum_{j=0}^{M-1} A_{k-1}(i) G_k(i, j) B_k(j)} \tag{9}$$

where

$$A_k(i) = \sum_{j=0}^{M-1} A_{k-1}(j) G_k(j, i) \tag{10}$$

$A_1(i) = 1$ and

$$B_k(i) = \sum_{j=0}^{M-1} G_{k+1}(i, j) B_{k+1}(j) \tag{11}$$

$B_K(i) = 1$. By forward–backward algorithm, $A_k(i)$'s and $B_k(i)$'s are the probabilities of the ith sample at kth trellis started from the beginning and from the end of the trellis, respectively. In (9), the denominator is used for normalisation which makes $\sum_j C_k(j) = 1$.

In order to be used as one of the prior LLRs of (2) in the decoder, the sample probabilities $C_k(j)$'s need to be converted into bit probabilities. The sample-to-bit probabilities $s_n(v)$'s are the probabilities corresponding to the binary notation of the sample, which are given by

$$s_n(v) = \frac{\sum_{0 \le j \le M,\ u_l(j) = v} C_k(j)}{\sum_{0 \le i \le M} C_k(i)} \tag{12}$$

where $n = (k-1)L + l$ and $v \in \{0, 1\}$. Let $O_{S,n}$ be LLR of the sample-to-bit probabilities, that is, $O_{S,n} = \ln(s_n(1)/s_n(0))$. Then, the extrinsic LLRs $O_{\mathrm{SE},n}$'s of the MC module are given, as in (4), by

$$O_{\mathrm{SE},n} = O_{S,n} - O_{\mathrm{CE},n} \tag{13}$$

for $1 \le n \le N_m$.

This whole process of the MC module is summarised as follows.

Algorithm 1: MC module

Input: $O_{\mathrm{CE},n}$, a_{ij}
Output: $O_{\mathrm{SE},n}$
Step 1: Calculate the bit-to-sample probabilities $g_k(j)$'s from the extrinsic LLRs $O_{\mathrm{CE},n}$'s on message bits by (7).
Step 2: Generate the sample-transition gains $G_k(i, j)$'s, using (8).
Step 3: Calculate sample probabilities $C_k(j)$'s at the kth trellis using forward–backward algorithm with the sample-transition gains $G_k(i, j)$'s, as in (9).
Step 4: Evaluate the sample-to-bit probabilities $s_n(v)$'s of (12) from the sample probabilities $C_k(j)$'s.
Step 5: Obtain the LLRs of the MC module, $O_{S,n}$'s, from the sample-to-bit probabilities $s_n(v)$'s.
Step 6: Calculate extrinsic LLRs $O_{\mathrm{SE},n}$'s of the MC module from the LLRs $O_{S,n}$'s by (13).

4 Simplified MC module

In this section, a simplified MC module is proposed to alleviate the computational complexity of the MC module in (9)–(11). The M-state MC for non-binary source samples can be replaced with the $\log_2 M$ number of two-state MCs. The M-state MC is SMM and the $\log_2 M$ number of two-state MCs are BMM. We observe that each bit in a sample also preserves a certain degree of correlation when the sample sequence is correlated. Thus, each two-state MC of BMM is employed on each bit of a sample.

The computational load of the BMM module is lower than that of the SMM module, especially when the number of states M is large. In order to decode K number of source samples, the computational complexity of the SMM module is $O(KM^2)$ [15]. In SMM, a current sample is designated to one of M candidate states with a certain transition probability. Since all the states are connected with each other, in general, in a SMM trellis section, the number of connections is proportional to the square of the number of states. On the other hand, the computational complexity of BMM is only $O(K \log_2 M)$. It is because BMM consists of $\log_2 M$ number of two-state MCs and the number of connections in a single trellis section of a two-state MC is 2^2, regardless of the number of state M.

To apply BMM instead of SMM, the transition probabilities of BMM need to be estimated. The transition probabilities of the two-state MC for the lth bit, denoted by $q_{l,ij}$, $1 \le l \le L$, $0 \le i, j \le 1$, are obtained based on that of the M-state MC as follows (see (14))

$$\begin{aligned}
q_{l,ij} &\overset{(a)}{=} \Pr[b_{2,l} = j | b_{1,l} = i] \\
&\overset{(b)}{=} \sum_{b_{1,1}} \cdots \sum_{b_{1,(l-1)}} \sum_{b_{1,(l+1)}} \cdots \sum_{b_{1,L}} \frac{\Pr[b_{2,l} = j, b_{1,1}, \ldots, b_{1,(l-1)}, b_{1,l} = i, b_{1,(l+1)}, \ldots, b_{1,L}]}{\Pr[b_{1,l} = i]} \\
&\overset{(c)}{=} \sum_{\substack{w=0, \\ u_l(w) = i}}^{M-1} \frac{\Pr[b_{2,l} = j, x_1 = w]}{\Pr[b_{1,l} = i]} \\
&\overset{(d)}{=} \sum_{\substack{w=0, \\ u_l(w) = i}}^{M-1} \frac{\Pr[b_{2,l} = j | x_1 = w] \Pr[x_1 = w]}{\Pr[b_{1,l} = i]}
\end{aligned} \tag{14}$$

where $u_l(w) = i$ means the lth bit in the binary representation of the state w is equal to i, (a) holds by Markov property, the theorem of total probability [19] is applied in (b) and (c), and the binary representation of the second equation is expressed

as a sample in the third equation, that is, $x_1 = [b_{1,1}, \ldots, b_{1,L}]^{\mathrm{T}} = w$. In (d) of (14), $\Pr[b_{2,l} = j | x_1 = w]$ can be calculated from the transition probabilities of the M-state MC, that is

$$\Pr[b_{2,l} = j | x_1 = w] = \sum_{\substack{v=0,\\ u_l(v)=j}}^{M-1} a_{wv} \tag{15}$$

The denominator of (14), the stationary probability of the lth bit $\Pr[b_{1,l} = i]$, is obtained as the sum of the stationary probabilities of the corresponding states j where $u_l(j) = i$, which is given by

$$\Pr[b_{1,l} = i] = \sum_{\substack{j=0,\\ u_l(j)=i}}^{M-1} \Pr[x_1 = j] \tag{16}$$

Note that (14) and (16) are the transition probabilities and the stationary probabilities of the lth two-state MC. The same procedure can be repeated for each $1 \le l \le L$ to build $L(= \log_2 M)$ two-state MCs. Thus, we have every component to construct BMM. Using (14)–(16), the MC module can be simplified to the BMM module.

We note that using the BMM module leads to degradation in performance. In the trellis of SMM, the number of possible transition paths is M^2, whereas in BMM it is $2^2 \log_2 M$. Since SMM considers more transition paths than BMM ($M^2 > 2^2 \log_2 M$ for $M > 2$), replacing SMM with BMM can cause reduction of prior information. This makes it interesting to study how much reduction one should expect to see.

To justify the usefulness of BMM, an analysis based on comparison of entropy rates, one for SMM and the other for the corresponding BMM, is carried out. The entropy rate of an MC measures the amount of uncertainty per sample [1]. When the entropy rate of BMM is close to that of SMM, we can safely say, BMM is equally capable of representing the correlated source. Thus, we aim to use the entropy rate as a criterion for closeness between the two models. This can give us useful insights before carrying out extensive computer simulations to determine the performance difference.

The entropy rate of an MC can be calculated if two quantities, the stationary distribution and the transition probability, are available. They are obtained in (14) and (16) for BMM and the transition probability of SMM is assumed to be known and the stationary distribution of SMM is obtained from the transition probability [20]. Thus, the entropy rate of SMM is given by

$$\begin{aligned} H(x_2|x_1) &= \sum_{i=0}^{M-1} \Pr[x_1 = i] H(x_2|x_1 = i) \\ &= \sum_{i=0}^{M-1} \Pr[x_1 = i] \left(- \sum_{j=0}^{M-1} a_{ij} \log a_{ij} \right) \end{aligned} \tag{17}$$

where $\Pr[x_1 = i]$ is the stationary probability of the ith state. The entropy rate of BMM is the sum of the entropies of $\log_2 M$ separated two-state MCs. The entropy rate of SMM and that of BMM are compared as follows

$$\begin{aligned} H(x_2|x_1) &\overset{(a)}{=} H(b_{2,L}|b_{2,(L-1)}, \ldots, b_{2,1}, b_{1,1}, \ldots, b_{1,L}) + \cdots \\ &\quad + H(b_{2,1}|b_{1,1}, \ldots, b_{1,L}) \\ &\overset{(b)}{\le} \sum_{l=1}^{L} H(b_{2,l}|b_{1,l}) \end{aligned} \tag{18}$$

where (a) follows from chain rules and (b) is due to the fact that more conditioning is to reduce its entropy, and the equality holds iff $H(b_{2,l}|b_{2,(l-1)}, \ldots, b_{2,1}, b_{1,L}, \ldots, b_{1,l}, \ldots, b_{1,1}) = H(b_{2,l}|b_{1,l})$ for $1 \le l \le L$.

The entropy rate of BMM is bigger than or equal to that of SMM as represented by (b) in (18). The difference, from the entropy rate of SMM to that of BMM, exists and is always positive since the smaller number of transition paths are used in BMM than SMM. This difference can be considered as an indication of model mismatch from SMM to BMM while it is very difficult to predict the expected performance degradation from using this mismatched model analytically. For example, if the difference is small, we can expect, use of the simplified model, BMM, would work as effectively as that of the full complexity SMM.

5 Simulation results

We evaluate BER performance difference of the two modules, the SMM against the BMM, in extensive Monte Carlo computer simulation. For simulation, correlated non-binary source samples are generated from a stationary M-state MC in the following manner: the transition probability to remain in the same state is assumed to be $1 - p$, where p is a transition parameter, $0 < p < 1$, that is, $\Pr(x_k = i | x_{k-1} = i) = 1 - p$ for all states $0 \le i \le M - 1$; the probability of making transition to one of its neighbouring states from the previous state $1 \le i \le M - 2$ is $p/2$, that is, $\Pr(x_k = i + 1 | x_{k-1} = i) = \Pr(x_k = i - 1 | x_{k-1} = i) = p/2$ and that from the previous state $i = 0$, $M - 1$ to the neighbouring state is p, that is, $\Pr(x_k = 1 | x_{k-1} = 0) = \Pr(x_k = M - 2 | x_{k-1} = M - 1) = p$. The smaller p implies the higher correlation between the source samples. For simulation, we assume $M = 16$, that is, a 16-state MC as SMM is considered. Thus, BMM is four 2-state MCs. As a channel code, a ($N = 1024$, $N_m = 512$, $d_v = 4$, $d_c = 8$) regular LDPC code is used [21]. BER performance improves with increase in super-iterations, but the amount of improvement in each additional super-iteration decreases. We fix two super-iterations throughout simulation, and ten internal LDPC decoding iterations per super-iteration. The final decoded bit sequence $\hat{\boldsymbol{b}}$ is decided after the iteration is completed.

By evaluating the entropy rates of the source samples with varying p, the amounts of statistical information that is representable with a source model, SMM and BMM, can be

Table 1 Entropy rates of SMM and BMM ($M = 16$) for various transition probability p

p	$H(x_2\|x_1)$ of SMM (*i.* bits/sample)	$H(x_2\|x_1)$ of BMM (*i.* bits/sample)	Difference of the entropy rates of BMM and SMM
0.05	0.333	0.559	0.226
0.10	0.562	0.940	0.378
0.20	0.909	1.514	0.605

doi: 10.1049/iet-com.2011.0840

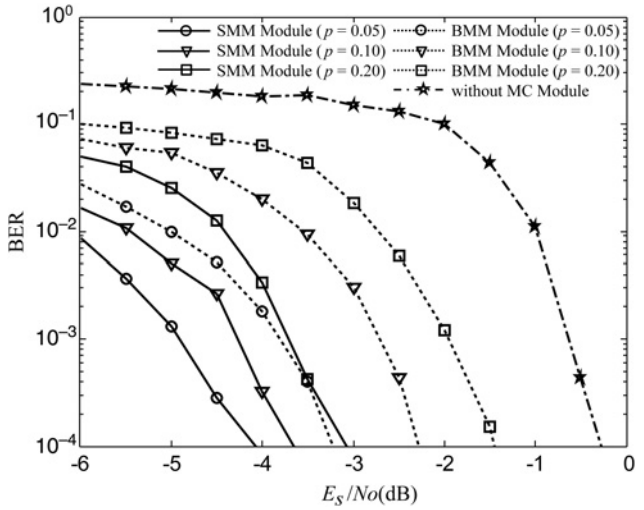

Fig. 3 *BER comparisons for the SMM and the BMM module ($M = 16$)*

estimated. For example, when $p = 0.05$, the entropy rate of the source sample is 0.333 (*i.* bits/sample), where *i.* bits means information bits. The entropy rate of SMM is also 0.333 (*i.* bits/sample) under the assumption that the SMM module extracts the perfect statistical information of the source samples. On the other hand, the entropy rate of BMM with transition probabilities $q_{l,ij}$ as in (14) is estimated to be 0.559 (*i.* bits/sample). Table 1 shows the entropy rates of SMM and BMM with varying p. As shown in Table 1, as p decreases, the correlation of source samples is increased and the difference of the entropy rates of the two models narrows.

Fig. 3 shows comparison in terms of BER performance of the two modules. As we expected from the comparison of the entropy rates, BER performance of the two is improved as p gets smaller. When $p = 0.05/0.10/0.20$, performance of the joint source–channel decoding scheme with the SMM module is improved by 3.8/3.4/2.8 dB in SNR; whereas the one with the BMM is 3.0/2.0/1.2 dB in SNR compared to the BER performance of the plain channel decoding scheme, that is, without any MC module, measured at BER $= 10^{-4}$. The cooperation results with the BMM module are shown sensitive to the values of p, compared to those with the SMM module. When samples are less correlated, loss of prior information obtained from the BMM module becomes more noticeable compared to the case of highly correlated samples. This is probably because of smaller number of the transition paths incorporated in BMM. Fortunately, the difference between the SMM module and the BMM module is getting reduced with smaller p. For highly correlated samples, the BMM module shows competitive BER performance with reduced computational complexity. The computational complexity of the SMM module is proportional to M^2. When the number of states M grows for finer representation of samples, therefore the amount of computations for the SMM module is drastically increased. However, the computational load of the BMM module is only proportional to $\log_2 M$. Therefore the BMM module can be applied to the cases where uncompressed source samples show a certain degree of correlation, while incurring much smaller complexity to the receiver.

Theoretical capacity curves as a function of SNRs given the code rate and the constellation size would be useful to give an asymptotic limit of the proposed system. As we aim at checking whether the BMM module can be used as a complexity reduction alternative to the SMM module at a finite block length, we do not include capacity results in this paper. Instead, interested readers are referred to the capacity results given in [22].

6 Conclusions

In the joint source–channel decoding, as the number of states of the Markov source model increases, the receiver incurs computational load of $O(KM^2)$ to update the probabilities used in the forward–backward algorithm. To mitigate this problem, we have proposed the BMM module that employs $\log_2 M$ two-state MCs, instead of single M-state MC. Since the number of branches in the trellis of BMM is much smaller than that of SMM, the computational complexity for forward–backward algorithm can be reduced to $O(K \log_2 M)$. Simultaneously, reduction in the number of transition paths in BMM leads to degradation of the prior information obtained from cooperation with the MC module. BER performance is expected to be degraded using BMM. However, as anticipated from our observation on the comparison of the entropy rates of BMM and SMM, it was shown in simulation that when the transition parameter p is small, the decoder with the BMM module can achieve more gain than when the transition parameter p is large. Simulation results demonstrate the BMM module can lead to reduction in the required SNR.

7 Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MEST) (Do-Yak Research Program, NO. 2011-0016496). Part of materials in this paper has been presented at the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society in 2007.

8 References

1 Shannon, C.E.: 'A mathematical theory of communication', *Bell Syst. Tech. J.*, 1948, **27**, pp. 379–423, 623–656
2 Vembu, S., Verdú, S., Steinberg, Y.: 'The source–channel separation theorem revisited', *IEEE Trans. Inf. Theory*, 1995, **41**, (1), pp. 44–54
3 Fresia, M., Peréz-Cruz, F., Poor, H.V., Verdú, S.: 'Joint source and channel coding', *IEEE Signal Process. Mag.*, 2010, **27**, (6), pp. 104–113
4 Xiong, Z., Liveris, A.D., Cheng, S.: 'Distributed source coding for sensor networks', *IEEE Signal Process. Mag.*, 2004, **21**, (5), pp. 80–94
5 Hagenauer, J.: 'Rate-compatible punctured convolutional codes (RCPC codes) and their applications', *IEEE Trans. Commun.*, 1988, **36**, (4), pp. 389–400
6 Sayood, K., Borkenhagen, J.C.: 'Use of residual redundancy in the design of joint source/channel coders', *IEEE Trans. Commun.*, 1991, **39**, (6), pp. 838–846
7 Wang, Y., Zhu, Q.: 'Error control and concealment for video communication: a review', *Proc. IEEE*, 1998, **86**, (5), pp. 974–997
8 Kim, M., Lee, S., Kim, S.: 'Spatial error concealment method based on POCS with a correlation-based initial block', *IET Image Process.*, 2007, **1**, (2), pp. 134–140
9 Xu, Y., Zhou, Y.: 'Adaptive temporal error concealment scheme for H.264/AVC video decoder', *IEEE Trans. Consum. Electron.*, 2008, **54**, (4), pp. 1846–1851
10 Garcia-Frias, J., Villasenor, J.D.: 'Joint turbo decoding and estimation of hidden Markov sources', *IEEE J. Sel. Areas Commun.*, 2001, **19**, (9), pp. 1671–1679
11 Zamani, M., Lahouti, F.: 'Distributed source coding using symbol-based and non-binary turbo codes – applications to wireless sensor networks', *IET Commun.*, 2008, **2**, (8), pp. 1089–1097

12 Liveris, A.D., Xiong, Z., Georghiades, C.N.: 'A distributed source coding technique for correlated images using turbo-codes', *IEEE Commun. Lett.*, 2002, **6**, (9), pp. 379–381
13 Zhao, Y., Garcia-Frias, J.: 'Joint estimation and compression of correlated nonbinary sources using punctured turbo codes', *IEEE Trans. Commun.*, 2005, **53**, (3), pp. 385–390
14 Yao, N., Lee, H., Chang, C., Sclabassi, R.J., Sun, M.: 'A power-efficient communication system between brain-implantable devices and external computers'. Proc. 29th Int. Conf. IEEE Engineering in Medicine and Biology Society (EMBS), Lyon, France, August 2007, pp. 6588–6591
15 Khreich, W., Granger, E., Miri, A., Sabourin, R.: 'On the memory complexity of the forward–backward algorithm', *Pattern Recognit. Lett.*, 2010, **31**, (2), pp. 91–99
16 Gallager, R.: 'Low-density parity-check codes', *IRE Trans. Inf. Theory*, 1962, **8**, (1), pp. 21–28
17 Baum, L.E., Petrie, T., Soules, G., Weiss, N.: 'A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains', *Ann. Math. Stat.*, 1970, **41**, (1), pp. 164–171
18 Berrou, C., Glavieux, A.: 'Near optimum error correcting coding and decoding: turbo-codes', *IEEE Trans. Commun.*, 1996, **44**, (10), pp. 1261–1271
19 Stark, H., Woods, J.W.: 'Probability and random processes with applications to signal processing' (Prentice-Hall, 2002, 3rd edn.)
20 Cover, T.M., Thomas, J.A.: 'Elements of information theory' (John Wiley and Sons, 2006, 2nd edn.)
21 Yue, G., Lu, B., Wang, X.: 'Analysis and design of finite-length LDPC codes', *IEEE Trans. Veh. Technol.*, 2007, **56**, (3), pp. 1321–1332
22 Yao, N., Lee, H., Sclabassi, R.J., Sun, M.: 'Low power digital communication in implantable devices using volume conduction of biological tissues'. Proc. 28th Int. Conf. IEEE Engineering in Medicine and Biology Society (EMBS), New York, USA, August 2006, pp. 6249–6252

doi: 10.1049/iet-com.2011.0840