Adaptive Slepian-Wolf Decoding Based On Expectation Propagation

Lijuan Cui, Shuang Wang and Samuel Cheng

Abstract—A major difficulty that plagues the practical use of Slepian-Wolf (SW) coding (and distributed source coding in general) is that the precise correlation among sources needs to be known a priori. However, belief propagation (BP) algorithm cannot adapt efficiently to the statistical change of the correlation. This paper proposes an adaptive SW decoding scheme which can perform online time-varying correlation estimation at the bit-level by incorporating expectation propagation (EP) algorithm. Moreover, we compare the proposed EP-based approach with Monte Carlo method using particle filtering (PF) algorithm. Our results show that the proposed EP estimator obtains the comparable estimation accuracy with less computational complexity than the PF method.

Index Terms—Adaptive decoding, Distributed algorithms, Source coding, Data compression

I. INTRODUCTION

Slepian-Wolf (SW) coding is a technique to losslessly compress correlate remote sources separately and decompress them jointly [1]. Numerous channel coding based SW coding schemes have been proposed [2], [3]. However, the fundamental assumption is that the correlation statistics needs to be known accurately a priori. Actually in many real-world applications, such as sensor networks, the correlation statistics among sensors cannot be obtained easily and may vary over both space and time. Since the decoding performance of distributed source coding (DSC) relies on the knowledge of correlation significantly, the design of an online correlation estimation scheme becomes an important research topic both in theoretical study and practical applications [4]–[6].

Here we consider asymmetric SW coding of two binary correlated sources X and Y, where the relationship is modeled as a virtual binary symmetric channel (BSC) with a time-varying crossover probability. For the crossover probability estimation, many algorithms were studied in the literature. In [4], the residual redundancies in LDPC syndromes are used to estimate the crossover probability between two correlated binary sources using Mean-Intrinsic-LLR. However, this algorithm works only for highly correlated sources. In [5], [6], the expectation maximization (EM) algorithm was used to estimate the crossover probability. However, the crossover probability is assumed to be constant and does not change within each codeword block. In [7] we considered adaptive correlation estimation with a single factor graph. The algorithm can handle sources with both weak and strong correlations and the statistics may vary dynamically. However, since the correlation parameter is continuous and cannot be parametrized in general, we incorporated Monte Carlo step into standard BP. The resulting particle based BP (PBP) algorithm handles correlation estimation well but with significant computational overhead as the computational complexity of BP algorithm increases exponentially with the alphabet size.

In this letter, we consider another possible workaround for approximate inference. Instead of using Monte Carlo techniques, we explore the possibility of deterministic approximation, in particular, through the use of expectation propagation (EP). Comparing to Monte Carlo techniques, deterministic approximation typically is much faster but is less flexible. It may not work for all problems and is also more mathematically involved. In the following, we demonstrate how EP can be used for adaptive SW decoding. Moreover, we compare the performance of the EP estimator with PBP estimator [7]–[9] under the same setup. Our simulation results show that the proposed EP estimator obtains the comparable estimation accuracy with less computational complexity comparing to PBP. Further, the EP estimator does not depend on the initial estimation of crossover probability and offers a good real-time estimation for the crossover probability. As a result, a lower decoding error rate is possible comparing to a standard BP-based SW decoder.

II. FACTOR GRAPH CONSTRUCTION OF SW DECODER WITH CROSSOVER PROBABILITY ESTIMATION

A factor graph for both SW decoding and correlation tracking is illustrated in Fig. 1. The factor graph is more or less the same as that used in our prior work [7]. Note that regions II and III in Fig. 1 alone contribute the same factor graph as that used in traditional LDPC-based SW decoding. A block of X (X1, X2, ···, XN) is compressed into M syndrome bits s1, s2, ···, sM, thus resulting in M : N compression. Here, x1 and y1 are realizations of nodes X1 and Y1, respectively. In Region III, syndrome factor nodes hM, k = 1, 2, ···, M take into account the constraints imposed by the received syndrome bit sk. In Region II, factor node fi plays a role of providing a predetermined likelihood p(yi|x1, ρ) to variable node Xi for LDPC-based SW decoding, where ρ denotes the crossover probability. According to the relationship between xi and yi in BSC, the corresponding factor function of fi is defined as

\[ f_i(\rho, y_i) = \rho^{x_i \oplus y_i} (1 - \rho)^{1 \oplus x_i \oplus y_i}, \]

where \( \oplus \) is the bitwise sum of two elements.

However, in reality, crossover probability may vary over time, denoted by \( \rho_t \), and the perfect knowledge of time-varying crossover probability may not always be available at the decoder. Thus, in the case without feedback channels, it is necessary to perform an online crossover probability estimation to avoid the degradation of decoding performance. It also means that each factor node fi will periodically update the likelihood p(yi|x1, \( \rho_t \)) for the respective bit variable node Xi when a new crossover probability estimate of \( \rho_t \) is available, instead of using a predetermined likelihood p(yi|x1, \( \rho \)).
To enable the online estimation of time-varying crossover probability $\rho_i$, we introduce extra variable nodes $P_j$ and factor nodes $G_j$, $j = 1, 2, \ldots, N'$ (see Fig. 1 of the following section). Here, we call the number of factor nodes in Region I connecting to each variable node $P_j$ the connection ratio $C$ \footnote{To estimate a stationary crossover probability, we can set the connection ratio equal to the code length.}, which is equal to four in Fig. 1. In Region I, each variable node $P_j$ is used to model the time-varying crossover probability $\rho_i$ of a block of $C$ number of code bits. Moreover, the factor function $G_j(\rho_j)$ of factor node $G_j$ corresponds to the a priori distribution for variable $\rho_j$ and will be discussed later. Consequently, by introducing crossover probability estimation in Region I, likelihood function (1) will be updated to

$$f_i(\rho_j, x_i, y_i) = \rho_j^{y_i} (1 - \rho_j)^{1-y_i}.$$  

(2)

III. POSTERIOR APPROXIMATION ON FACTOR GRAPH

In Bayesian inference, the estimation of crossover probability $\rho_j$ corresponds to the estimation of its posterior distribution, i.e. $p(\rho_j|y_j)$, where $y_j = (y_i)_{i \in N^0(P_j)}$ and $N^0(P_j)$ represents the set of all neighbors’ indices for a variable node $P_j$ except the index of $G_j$. According to the Bayes’ rule, the posterior distribution over variable $\rho_j$ in Fig. 1 can be written as

$$p(\rho_j|y_j) = \frac{1}{Z_j} \prod_{i \in N^0(P_j)} p(\rho_j)p(y_i|\rho_j)$$  

(3)

$$= \frac{1}{Z_j} \prod_{i \in N^0(P_j)} \left( \int_{x_i} p(\rho_j)p(x_i)p(y_i|x_i, \rho_j) \right),$$  

where $Z_j = \int_{\rho_j} \prod_{i \in N^0(P_j)} p(\rho_j)p(y_i|\rho_j)$ is a normalization constant, $p(\rho_j) = g(\rho_j)$, $p(y_i|x_i, \rho_j) = f(\rho_j, x_i, y_i)$, the a priori distribution $p(x_i)$ is captured by the message $m_{X_i \rightarrow f_j}(x_i)$, and $m_{X_i \rightarrow f_j}(x_i)$ has two terms and the product of all the messages

$$\prod_{i \in N^0(P_j)} m_{X_i \rightarrow f_j}(\rho_j)$$  

has total $2^C$ number of terms, where $C = |N^0(P_j)|$ is the connection ratio, and it can be a large number (e.g. 50 to 10,000 in our study). To solve this problem, we resort to EP algorithm to approximate the posterior in the following section.

A. Posterior Approximation Using EP

Briefly speaking, EP algorithm attempts to seek an approximate posterior distribution restricted in exponential family that can be very close to the true posterior distribution through minimizing the Kullback-Leibler (KL) divergence between the true distribution and the approximation \cite{11}. In our problem, we assume an approximation to $p(\rho_j|y_j)$ of (4) in the form $q(\rho_j|y_j) = \frac{1}{\sum_{\rho_j} m_{G_j \rightarrow P}(\rho_j) \prod_{i \in N^0(P_j)} m_{f_j \rightarrow P}(\rho_j)}$, where each original message $m_{f_j \rightarrow P}(\rho_j)$ in (4) is replaced by an approximating message $\tilde{m}_{f_j \rightarrow P}(\rho_j)$ belonging to a tractable distribution in exponential family. The approximation of each message $\tilde{m}_{f_j \rightarrow P}(\rho_j)$ is achieved by minimizing KL divergence in the context of all the remaining messages.

Note that Beta distribution is defined as

$$\text{Beta}(x, \alpha, \beta) = \frac{1}{\text{beta}(\alpha, \beta)} x^{\alpha-1}(1-x)^{\beta-1}, \quad (5)$$

where $\alpha$ and $\beta$ are shape parameters, $\text{beta}(\alpha, \beta)$ represents Beta function. By comparing (2) with Beta distribution (5), likelihood function (2) can be represented in terms of Beta distribution with respect to variable $\rho_j$ as parameter:

$$f_i(\rho_j, x_i, y_i) = \text{Beta}((x_i + y_i) + 1, (1 + x_i + y_i) + 1) \times \text{Beta}(\rho_j, (x_i + y_i) + 1, (1 + x_i + y_i) + 1). \quad (6)$$

The original EP algorithm proposed by Minka is applied to approximate a mixture of Gaussian distributions in clutter problem \cite{11}. In this paper, our problem is to estimate time-varying crossover probability given observations from neighboring factor nodes (see (3) and (6)). Thus, EP is extended to approximate a mixture of Beta distributions instead of Gaussian distributions in our problem.

In addition, to approximate a posterior distribution of crossover probability, it is computationally favorable to choose a conjugate prior for the likelihood function based on Bayesian theorem. Since Beta distribution is the conjugate prior for itself, we choose $g(\rho_j) = \text{Beta}(\rho_j, \alpha^0_j, \beta^0_j)$ as the prior distribution with the prior parameter $\alpha^0_j$ and $\beta^0_j$.

EP algorithm processes in the following. For the ease of exposition, we denote by $q(\rho_j)$ the approximated posterior distribution instead of $\tilde{q}(\rho_j|y_j)$ in the rest of this letter.

1. Initialize the prior messages for the crossover probability variables

$$g_i(\rho_j) = \text{Beta}(\rho_j, \alpha^0_j, \beta^0_j) = \zeta_j^{\alpha^0_j-1}(1 - \rho_j)^{\beta^0_j-1}, \quad (7)$$

with $\alpha^0_j = 2$, $\beta^0_j = \frac{\alpha^0_j-1}{\rho^0_j} + 2 - \alpha^0_j$ and $\zeta_j = \frac{1}{\text{Beta}(\alpha^0_j, \beta^0_j)}$, where $\rho^0_j$ is the initial crossover probability for SW decoding, $\beta^0_j$ and $\alpha^0_j$ are shape parameters for Beta distribution.

2. Initialize the likelihood messages from the channel output

$$\tilde{m}_{f_j \rightarrow P}(\rho_j) = z_{ij} \rho_j^{\alpha^0_j-1}(1 - \rho_j)^{\beta^0_j-1} \quad (8)$$

with $\beta_{ij} = 1$, $\alpha_{ij} = 1$ and $z_{ij} = 1$, where the values selection for the above parameters guarantee that $\rho_j$ in (8) is equality likely before LDPC decoding.
The EP/PBP estimator starts working after 50 BP iterations and then is performed periodically every 20 BP iterations until BP decoder successfully decodes the codeword or reaches its maximum number of iterations (i.e. 150 in our experiment). From Fig. 3(a), we can see that there is no obvious difference of performance between the proposed EP based decoder and PBP decoder for constant crossover probability, and we observe a small performance gap between EP based decoder and PBP decoder for time-varying crossover probability in Fig. 3(b). On the other hand, the decoding performance of SW decoders with EP/PBP estimator are significantly better than that of the standard BP-based SW decoder in both Fig. 3(a) and (b). It is important to point out that the additional computational overhead of the proposed EP based decoder is less than 10\% of the standard BP-based SW decoder. In the following comparison, the proposed EP and PBP algorithms are implemented in MATLAB with Java, and executed on an intel i7 CPU with 100 trails. In our simulation, EP based decoder requires about 68s, while PBP decoder needs 814s. We can see that EP based decoder is much faster than PBP based decoder. Thus, we believe that proposed SW decoder with EP estimator is a practical improved alternative of the standard BP-based SW decoder and PBP decoder.

REFERENCES