Decoder driven adaptive distributed arithmetic coding

DECODER-DRIVEN ADAPTIVE DISTRIBUTED ARITHMETIC CODING

Marco Grangetto

Università degli Studi di Torino (Italy)
Dipartimento di Informatica
marco.grangetto@polito.it

Enrico Magli, Gabriella Olmo

Politecnico di Torino (Italy)
Dipartimento di Elettronica
enrico.magli@polito.it

ABSTRACT

We propose a distributed source coding system for data collected by sensor networks. It uses a feedback channel between the sensors and the gateway node (i.e., the joint decoder) but, unlike previous systems, the encoding process is driven by the decoder. Compression is performed using distributed arithmetic coding, which is extended to adaptively estimate the source probabilities. Specifically, the decoder estimates marginal and conditional probabilities, and sends them back to the sensors to drive the distributed arithmetic coding process. This reduces the decoding delay, and potentially eliminates the need of rate-compatible Slepian-Wolf codes.

Index Terms— Distributed source coding, arithmetic coding, sensor networks, adaptive arithmetic coding

1. INTRODUCTION AND MOTIVATION

Of late, a lot of attention has been devoted to the development of efficient communications for distributed scenarios. A typical system comprises several wireless battery-powered sensors that collect information about the environment; a gateway node collects the information from the sensors and processes it in order to extract the information of interest. The communication problem, however, is not trivial. The data collected by these sensors often exhibit significant statistical dependencies that should be exploited for compression. On the other hand, due to the limited energy resources, communications between sensors (other than with the gateway node) should be avoided; this makes it difficult to exploit such dependencies through proper encoder-based statistical modeling.

To deal with these problems, distributed source coding (DSC) techniques have recently been considered. In lossless DSC two (or more) statistically dependent information sources are compressed using separate encoders that are not allowed to talk to each other. DSC theory proves that separate encoding is optimal, provided that the sources are decoded jointly. Sensor 1 performs "standard" encoding of its data $\mathbf{Y}$ at a rate approximately equal to its entropy, and transmits it to the gateway node. Sensor 2 performs "conditional" encoding of its data $\mathbf{X}$ at a rate approximately equal to $H(\mathbf{X}|\mathbf{Y})$. DSC coders for the setting above are typically constructed using channel codes such as trellis codes, turbo codes and low-density parity-check (LDPC) codes [1]. More recently, distributed arithmetic coding (DAC) [2,3] has been proposed. In DAC, an arithmetic coder is modified in order to compress $\mathbf{X}$ at a rate below its entropy, generating a non uniquely decodable codeword; a sequential decoder reconstructs $\mathbf{X}$ given the side information $\mathbf{Y}$. This approach has good performance for short block length, enables to incorporate time-varying and contextual statistical information regarding the source, and provides a continuous set of rates.

Although much work has been done in the field of DSC, some problems still remain unsolved. One of them is that Sensor 2 needs to estimate $H(\mathbf{X}|\mathbf{Y})$ in order to select an appropriate coding rate for $\mathbf{X}$. In some DSC applications this is only partially a problem; if the two sources are co-located, as in distributed video coding, a few samples from the sources can be used to estimate the conditional entropy. However, this solution is not viable if $\mathbf{X}$ and $\mathbf{Y}$ are acquired by two physically separated sensors. Alternatively, one could use a feedback channel; using a rate-compatible DSC code, the decoder can request an increasingly refined description of $\mathbf{X}$ from Sensor 2, starting from rate zero and increasing a few bits at a time, until decoding is successful (see e.g. [4]). This procedure leads to correct decoding, but can introduce delays because of the many encoder/decoder interactions and the multiple decoding attempts; moreover, it requires knowledge of the conditional probability distribution $P(\mathbf{X}|\mathbf{Y})$ at the decoder.

In this paper we propose a system that employs a feedback channel, but is totally driven by the decoder. In particular, the joint decoder computes on-the-fly estimates of the statistics of $\mathbf{X}$ and $\mathbf{X}|\mathbf{Y}$, and instructs Sensor 2 on the coding rate that it has to use for the subsequent symbols until the next update. The proposed scheme is based on DAC; the main contributions of this paper are the following. We upgrade DAC to adaptive distributed arithmetic coding (ADAC), where, unlike [2], we do not assume any knowledge of the *a priori*

This work has been supported in part by EU grant "SEA - Seamless Content Delivery", and in part by EU Network of Excellence "NewCom++".

source statistics. We propose a method that exploits statistical soft information available at the sequential joint decoder to estimate the prior probabilities of $\mathbf{X}$ during the decoding process, with a small delay. We use this information to select a suitable coding rate at the encoder. In this way, although Sensor 2 and the gateway node still communicate with each other through the feedback channel, all communications are done on-the-fly during the encoding of $\mathbf{X}$. A single decoding takes place, as opposed to the multiple decoding attempts performed by current systems. As a consequence, the delay is greatly reduced. Moreover, the on-the-fly adaptation of the statistical model eliminates the need for a rate-compatible code, because the adaptation of the coding rate ideally leads to correct decoding at the first attempt.

2. PROPOSED METHOD

2.1. Review of distributed arithmetic coding

Let $\mathbf{X} = [X_0, X_1, \ldots, X_i, \ldots, X_{N-1}]$ be a length-N binary i.i.d. symbol sequence with probabilities $p_0 = P(X_i = 0)$ and $p_1 = P(X_i = 1)$, and $\mathbf{Y} = [Y_0, Y_1, \ldots, Y_i, \ldots, Y_{N-1}]$ be a side information binary i.i.d. symbol sequence. The standard arithmetic coding process maps the source symbols onto sub-intervals of $[0, 1)$ using recursive interval selection. For each input symbol X_i, the coding interval is partitioned into two adjacent sub-intervals whose lengths are proportional to estimated values of p_0 and p_1; the sub-interval representing X_i is selected for the next iteration. The codeword $\mathbf{C}_X$ is a binary number lying in the final interval.

DAC inserts an ambiguity in the encoding process, using a modified interval subdivision strategy where the interval length is proportional to the modified probabilities $\widetilde{p}_0$ and $\widetilde{p}_1$. We set $\widetilde{p}_j = \alpha_j p_j$ for $j = 0, 1$, with $\alpha_j \geq 1$, so that $\widetilde{p}_0 + \widetilde{p}_1 \geq 1$. The recursive coding procedure is the same as in classical arithmetic coding, and yields a larger final interval containing a shorter codeword $\mathbf{C}_X$. In order to fit the enlarged sub-intervals $\widetilde{p}_j$ into the $(0, 1]$ interval, the sub-intervals are allowed to partially overlap. As a consequence, the decoder will generally be unable to decode the source without knowledge of the side information. Setting $\alpha_j = p_j^{-k}$, the average coding rate can be shown to be

$$R_X = (1-k) \sum_{j=0}^{1} p_j \log_2 \frac{1}{p_j} = (1-k)H$$

where H is the entropy of the probability model, i.e. $H = -\sum_{j=0}^{1} p_j \log_2 p_j$. The coding rate can be reduced to any desired value by means of a suitable selection of the parameter k. Setting $k = 0$ yields the classical arithmetic coder. More details on DAC can be found in [2].

At the decoder, $\mathbf{Y}$ is available without losses and is used to remove the ambiguity left in the codeword C_X. For brevity, in the following we only give a summary of the decoder operation; all the details can be found in [2]. In particular, the joint decoding process can be formulated as a symbol-driven sequential search along a proper decoding tree, where each node represents a state of the sequential arithmetic decoder of $\mathbf{X}$. We employ the *Maximum A Posteriori* (MAP) metric $P(\mathbf{X}|\mathbf{Y})$ and the M-algorithm in order to select the most-likely path in the tree, using $M = 2048$.

Given the correlation model in terms of the probability distribution $P(\mathbf{X}|\mathbf{Y})$, the MAP metric $P(X_0, \ldots, X_i|\mathbf{Y})$ can be evaluated iteratively during the decoding process as

$$P(X_0 \ldots, X_i|\mathbf{Y}) = P(X_0, \ldots, X_{i-1}|\mathbf{Y}) P(X_i|\mathbf{Y}) \quad (1)$$

In this paper we employ a memoryless correlation model, i.e. $P(X_i|\mathbf{Y}) = P(X_i|Y_i)$. As a consequence, each path in the tree is characterized by the metric $P(\mathbf{X}|\mathbf{Y}) = \prod_{i=0}^{N-1} P(X_i|Y_i)$ that can be efficiently evaluated in the log domain as a summation.

2.2. Adaptive DAC System

In the following we assume that $\mathbf{X}$ and $\mathbf{Y}$ are i.i.d. binary sources. The correlation model $P(X_i|Y_i)$ is assumed to be memoryless, as well. No prior estimates of $P(X_i)$, $P(Y_i)$ and $P(X_i|Y_i)$ are assumed to be available. The proposed ADAC system comprises the two encoders for $\mathbf{X}$ and $\mathbf{Y}$, and a joint decoder. The adaptive encoder for $\mathbf{Y}$ (i.e., the side information) operates with $k_Y = 0$ and employs the occurrence frequencies of 0 and 1 as estimates of $P(Y_i)$. In particular, the i-th symbol is encoded using the estimate [5]:

$$\widehat{P}(Y_i = y) = \frac{1 + \sum_{j=0}^{i-1} \mathbf{1}_{Y_j = y}}{2 + i}$$

where

$$\mathbf{1}_A = \begin{cases} 1, & \text{if } A \text{ is true} \\ 0, & \text{otherwise} \end{cases}$$

and $y = 0, 1$. In a such a way, $\mathbf{Y}$ is encoded adaptively and is transmitted without ambiguity. After an initial transient, the coding rate R_Y approaches the entropy $H(\mathbf{Y})$.

On the other hand, $\mathbf{X}$ must be encoded by ADAC exploiting the correlation with $\mathbf{Y}$. This requires to estimate $P(X_i)$ and to set the overlap parameter k_X depending on the amount of correlation between $\mathbf{X}$ and $\mathbf{Y}$. The probability $P(X_i)$ can be estimated in the same way as is done for the side information, because the sample values of $\mathbf{X}$ are available at Sensor 2. Since at the beginning of the encoding process no information is available at the decoder regarding the correlation, parameter k_X is initialized to 0 (i.e., non-distributed coding). This value is kept fixed for L samples, after which the encoder receives an updated parameter k_X from the decoder, as explained in the following.

The decoder needs to estimate the probability $P(X_i|Y_i)$ for the MAP decoding metrics, and the parameter k_X to be sent back to the encoder through the feedback channel. All estimations are done by dividing the original block of N bits into subblocks of length L.

2.2.1. Decoding of the first block

The decoder has perfect knowledge of the whole sequence $\mathbf{Y}$. Moreover, it can start decoding the first block $\mathbf{X}$ because no overlap has been used. However, the number of decoded symbols l is generally slightly inferior to the number of coded symbols L because, due to sequential encoding, the codeword bits are available with some delay, as the DAC encoder is not terminated at the end of each subblock, but only at the end of the sequence. In light of this, $P(X_i = x)$ can be estimated as

$$\widehat{P}(X_i = x) = \frac{1 + \sum_{j=0}^{l-1} \mathbf{1}_{X_j = x}}{2 + l} \tag{2}$$

with $x = 0, 1$. Then, $H(\mathbf{X})$ is approximated by

$$\widehat{H}(X) = -\sum_{x=0,1} \widehat{P}(X_i = x) \log_2 \widehat{P}(X_i = x) \tag{3}$$

The conditional statistics can be estimated with a similar approach:

$$\widehat{P}(X_i = x | Y_i = y) = \frac{1 + \sum_{j=0}^{l-1} \mathbf{1}_{X_j = x, Y_j = y}}{2 + \sum_{j=0}^{l-1} \mathbf{1}_{Y_j = y}} \tag{4}$$

The conditional entropy can be approximated by

$$\begin{aligned} \widehat{H}(\mathbf{X}|\mathbf{Y}) \quad = \quad & \sum_{x,y} \widehat{P}(X_i = x | Y_i = y) \widehat{P}(Y_i = y) \\ & \log_2 \widehat{P}(X_i = x | Y_i = y) \end{aligned} \tag{5}$$

The value of parameter k_X can be determined as

$$k_X = 1 - \gamma \widehat{H}(\mathbf{X}|\mathbf{Y}) / \widehat{H}(\mathbf{X}) \tag{6}$$

where $\gamma > 1$ generates an extra rate that compensates for the sub-optimality of the DAC decoder (i.e., we encode at a rate γ times larger than the estimated conditional entropy). The selected k_X yields a coding rate $R_X = \gamma \widehat{H}(\mathbf{X}|\mathbf{Y})$. Such k_X is sent to the DAC encoder, which uses it to encode the next subblock of L samples, determining a reduction of the coding rate that depends on the estimated correlation.

2.2.2. Decoding of the subsequent blocks

For all the other blocks, the DAC decoder receives an ambiguous description of the source and must resort to sequential MAP decoding. The metrics along the decoding tree are evaluated using the estimate $\widehat{P}(X_i|Y_i)$ of the previous block. The MAP decoder attempts to decode as many symbols as possible, up to symbol $l \leq mL$, where $m = 1, \ldots,$ is the number of already coded subblocks. When the decoder tree has reached symbol depth l, the probability estimates are updated based on the sequence that exhibits the best accumulated metric. Once these estimates are available, (3) and (4) are used to compute $\widehat{H}(X)$ and $\widehat{H}(X|Y)$, then k_X is computed using (6), and is sent to the encoder through the feedback channel. The encoder/decoder interactions are summarized in the following:

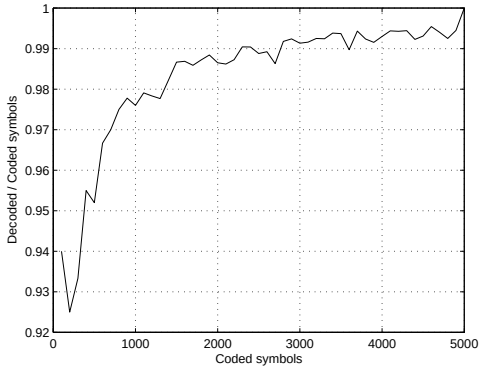

Fig. 1. Decoded/encoded symbols ratio versus number of encoded symbols.

1. $\mathbf{Y}$ is encoded without ambiguity.
2. Set $k_X = 0$ and $l = 0$.
3. While ($l < N$)
   - Encode next L symbols of $\mathbf{X}$ using DAC, and append more bits to the codeword C_X.
   - Joint decoder performs arithmetic decoding for the first subblock, or MAP decoding for the other subblocks, based on the partially received codeword, and decodes as many symbols as possible of the current subblock ($l \leq mL$).
   - Joint decoder updates its estimates $\widehat{P}(X_i)$ and $\widehat{P}(X_i|Y_i)$, and computes the value of k_X, which is returned to the encoder.

3. EXPERIMENTAL RESULTS AND DISCUSSION

The following experiments have been worked out for a source $\mathbf{X}$ with $P(X_i = 0) = P(X_i = 1) = 0.5$; the correlated side information $\mathbf{Y}$ is obtained transmitting $\mathbf{X}$ across a binary symmetric channel with a given crossover probability. This setting corresponds to $H(\mathbf{X}) = H(\mathbf{Y}) = 1$ bps. The crossover probability has been selected so that $H(\mathbf{X}|\mathbf{Y}) = 0.5$ bps.

In Fig. 1 the ratio between the number l of symbols available at the joint decoder and the number of coded source symbols mL, $m = 1, \ldots, \frac{N}{L}$ is reported as a function of the number of coded symbols. In this simulation $N = 5000$ symbols are coded with subblock size $L = 100$. The decoded/coded symbols ratio shows that, for the first block, about 94% of the subblock samples are decoded; for the second block, where overlapping is used, the percentage is slightly smaller, and quickly approaches 1, meaning that the delay between the encoder and the decoder is limited.

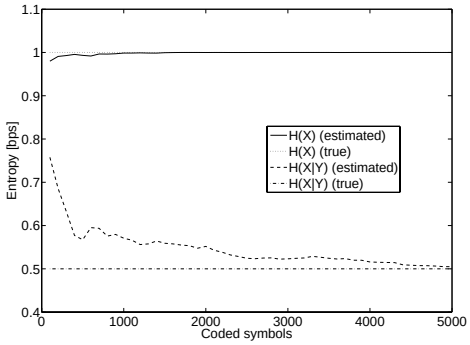

Fig. 2. Joint decoder entropy estimates.

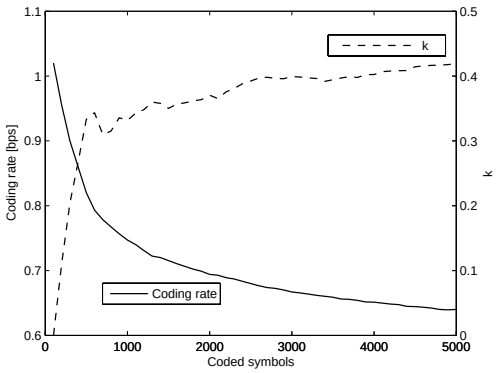

Fig. 3. Coding rate R_X and selected value of k_X versus the number of coded symbols.

Therefore, as the encoding proceeds, almost all the L source symbols, progressively coded during each encoder/decoder interaction, can be used by the decoder to update the coding parameters of DAC.

In Fig. 2 the entropy estimates $\widehat{H}(\mathbf{X})$ and $\widehat{H}(\mathbf{X}|\mathbf{Y})$ are shown versus the number of coded source symbols. The true values of such entropies are reported for comparison, and show that the ADAC is able to perform an accurate modeling of the observed correlation. In the present implementation of ADAC, the source and correlation models are assumed to be stationary over the entire coded block. Nevertheless, the modeling stage can be extended to the non-stationary case, by using a sliding window approach to evaluate time-varying statistical parameters.

In Fig. 3 the obtained values of k_X, with $\gamma = 1.15$, are shown as a function of the number of coded symbols. It can be seen that, in a few iterations, k_X gets close to the theoretical value 0.425, corresponding to the Slepian-Wolf coding bound of 0.5 bps plus the extra rate due to γ. In the same plot the average coding rate is reported as a function of the number of coded symbols. It can be seen that $N = 5000$ source symbols turn out to be coded on about 0.64 bps; the initial transient, due to the lack of prior correlation estimates, is evident in terms of the coding rate, as well. Finally, in Tab. 1 the performance of the proposed system is reported in terms of average coding rate, bit error rate (BER) and frame error rate (FER) for several values of γ. The reported figures are worked out by averaging the results of 1000 coding trials, employing $N = 2000$ and $L = 100$. As discussed in [2], the MAP decoding procedure does not guarantee error-free decoding; nevertheless, BER and FER can be reduced arbitrarily by admitting an increasing rate overhead through γ. For comparison, the non-adaptive DAC with exact knowledge of the source probability and the conditional entropy, achieves a BER equal to $4.92 \cdot 10^{-4}$ for block length $N = 1000$ and $\gamma = 1.30$.

For the setting considered in this paper, existing techniques based on turbo and LDPC codes cannot be applied directly, since adaptive versions of the related DSC coders have not been developed yet. Non-adaptive decoder-driven estimation of the conditional probability has been addressed in [6] for distributed video coding, but this is not directly comparable with the work presented in this paper.

Table 1. Performance in terms of R_X, BER and FER.

γ	R_X [bps]	BER	FER
1.05	0.60	$9.9\,10^{-2}$	$7.4\,10^{-1}$
1.15	0.66	$5.7\,10^{-2}$	$4.1\,10^{-1}$
1.30	0.72	$1.6\,10^{-2}$	$9.8\,10^{-2}$
1.50	0.81	$2.6\,10^{-3}$	$1.1\,10^{-2}$

4. REFERENCES

[1] Z. Xiong, A.D. Liveris, and S. Cheng, "Distributed source coding for sensor networks," *IEEE Signal Processing Magazine*, vol. 21, no. 5, pp. 80–94, Sept. 2004.

[2] M. Grangetto, E. Magli, and G. Olmo, "Distributed arithmetic coding," *IEEE Communications Letters*, vol. 11, no. 11, pp. 883–885, Nov. 2007.

[3] X. Artigas, S. Malinowski, C. Guillemot, and L. Torres, "Overlapped quasi-arithmetic codes for distributed video coding," in *Proceedings of IEEE ICIP*, 2007, pp. 9–12.

[4] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding," *Proceedings of the IEEE*, vol. Special Issue on Advances in Video Coding and Delivery, no. 1, pp. 71–83, Jan. 2005.

[5] I.H. Witten, R.M. Neal, and J.G. Cleary, "Arithmetic coding for data compression," *Communications of the ACM*, vol. 30, no. 6, pp. 520–540, June 1987.

[6] C. Brites, J. Ascenso, and F. Pereira, "Studying temporal correlation noise modeling for pixel based Wyner-Ziv video coding," in *Proceedings of IEEE ICIP*, 2006.