Efficient On-line Schemes for Encoding Individual Sequences with Side Information at the Decoder

Avraham Reani and Neri Merhav
Department of Electrical Engineering
Technion - Israel Institute of Technology
Technion City, Haifa 32000, Israel
Emails: [avire@tx,merhav@ee].technion.ac.il

***Abstract*—"THIS PAPER IS ELIGIBLE FOR THE STUDENT PAPER AWARD" We present adaptive on-line schemes for lossy encoding of individual sequences, under the conditions of the Wyner-Ziv (WZ) problem, i.e., the decoder has access to side information whose statistical dependency on the source is known. Both the source sequence and the side information consist of symbols taking on values in a finite alphabet $\mathcal{X}$. A set of fixed-rate scalar source codes with zero delay is presented. We propose a randomized on-line coding scheme, which achieves asymptotically (and with high probability), the performance of the best source code in the set, uniformly over all source sequences. The scheme uses the same rate and has zero delay. We then present an efficient algorithm for implementing our on-line coding scheme in the case of a relatively small set of encoders. We also present an efficient algorithm for the case of a larger set of encoders with a structure, using the method of the weighted graph and the Weight Pushing Algorithm (WPA). The complexity of these algorithms is no more than linear in the sequence length.**

I. Introduction

Consider a setting which has the following components: an individual source sequence, a discrete memoryless channel (DMC) with known statistics, a noiseless channel with rate constraint R, and a decoder. The encoder transforms the source sequence $x_1, x_2, \ldots, x_n$ into channel symbols $z_1, z_2, \ldots, z_n$, taking values in $\{1, 2, \ldots, M\}$, $M = 2^R$, which is transmitted to the decoder via a noiseless channel. The decoder, in addition to the encoded data arriving from the noiseless channel, has access to a side information sequence $y_1, y_2, \ldots, y_n$, created by transmitting the source sequence over the DMC. Using the data from the encoder and the side information, the decoder produces the reconstructed sequence $\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n$. The goal is to minimize the distortion between the source and the reconstructed signals by optimally designing an encoder-decoder pair. This is a variation of the problem of rate-distortion coding with decoder side information, which is well known as the WZ coding problem, first introduced in [6]. Like the ordinary rate-distortion problem, without side information, the problem of finding practical schemes that perform arbitrarily close to the rate-distortion curve is still largely open. The case of scalar source codes for the WZ problem was handled in several papers, e.g. [7] and [8]. In contrast to our case, these schemes operate under specific assumptions of known source statistics. WZ coding of a source with unknown statistics was also considered, e.g. in [9], and existence of universal schemes was established. However, these schemes are based on block coding or assume the knowledge of the source and side information sequences in advance, thus are irrelevant to the case of on-line encoding we handle.

A coding scheme is said to have an overall delay of no more than d if each channel symbol at time t, z_t, depends only on $x_1, \ldots, x_{t+d_1}$, each reconstructed symbol $\hat{x}_n$ depends only on $z_1, \ldots, z_{t+d_2}$ and $y_1, \ldots, y_{t+d_2}$, and $d \geq d_1 + d_2$. Weissman and Merhav [3], following Linder and Lugosi [2], constrcuted a randomized limited delay lossy coding scheme for individual sequences using methods based on prediction theory. These schemes perform, for any given set of source codes (called experts, this is the reference class), almost as well as the best source code in the set, for all individual sequences. The performance of the scheme is measured by the distortion redundancy, defined as the difference between the normalized cumulative distortion of the scheme and that of the best source code in the set, matched to the source sequence. The scheme is based on random choices of source codes from the set. The random choices are done according to exponential weights attached to each code. The weight of each sources code, each time we choose a code, depends on its past performance and has to be calculated. Thus, implementing this scheme for a large set of source codes requires efficient methods, to prevent prohibitive complexity. György, Linder and lugosy offered efficient algorithms for implementing this scheme for sets of scalar quantizers [4],[5] without side information.

In this paper, a fixed-rate, zero-delay adaptive coding schemes for individual sequences under the WZ conditions is presented. We define a set of scalar source codes for the WZ problem. Then, the scheme of [3] is extended for the case of the WZ problem w.r.t. the Hamming distortion measure. For any given set of WZ source codes, this scheme performs asymptotically as well as the best source code in the set, for all source sequences. We then demonstrate efficient implementations of this scheme. First, it is shown that the scheme can be implemented efficiently for any relatively small set of encoders, even though the set of decoders is large. Then, using graph-theoretic methods similarly to [5], we show that we can implement the scheme for large sets of scalar encoders with a structure. In the end, the implementations are generalized to any distortion measure, at the price of increased complexity.

It should be pointed out that the development of our efficient

on-line scheme is not a straightforward extension of those in [4][5] because of the following reasons: (i) because of the side information, the optimal partition of the source alphabet does not necessarily correspond to intervals. (ii) unlike in [4] and [5], for every given encoder in the reference class, there might be many possible decoders.

II. Definition of an on-line Adaptive WZ Scheme

Throughout this paper, for any integer n, we let a^n denote the sequence $a_1, a_2, \ldots, a_n$.

Given a source sequence x^n, the encoder transforms x^n into a sequence z^n whose symbols $\{z_i\}$ take on values in the set $\{1, 2, \ldots, M\}$. The decoder, in addition to z^n, has access to a sequence y^n, dependent on x^n via a known DMC, defined by the single-letter transition probability $P_{Y|X}(y_i|x_i)$ which is the probability of y_i, given x_i. Based on z^n and y^n, the decoder produces the reconstructed sequence $\hat{x}^n$. For convenience, we assume that x_i, y_i and $\hat{x}_i$ take on values in the same finite alphabet $\mathcal{X}$ with cardinality $|\mathcal{X}|$. All the results can be generalized straightforwardly to the case of different alphabets. The distortion between two symbols is defined to be the Hamming distortion:

$$\rho(x, \hat{x}) = \begin{cases} 0 & \text{if } x = \hat{x} \\ 1 & \text{elsewhere.} \end{cases} \tag{1}$$

We define the distortion for the input symbol at time t, x_t, as:

$$\Delta_t(x_t) = \mathbb{E}\rho(x_t, \hat{x}_t(z^t, y^t)) \tag{2}$$

where the expectation is taken with respect to y^t.

A. Definition of the reference set of encoders

In this part, we define the general set of scalar source codes, here referred to as experts. Each expert is a source code with a fixed rate, $R = \log M$. Practically, each expert partitions $\mathcal{X}$ into M disjoint subsets $(m_1, m_2, \ldots, m_M)$. The encoder e for each expert is given by a function $e : \mathcal{X} \rightarrow \{1, 2, \ldots, M\}$ that is, $z_i = e(x_i)$. The decoder d receives z_i, and together with the side information y_i, decides on $\hat{x}_i$, using a decoding function $d : \{1, 2, \ldots, M\} \times \mathcal{X} \rightarrow \mathcal{X}$, i.e., $\hat{x}_i = d(z_i, y_i)$. The definition above is not complete. It is easy to see that different encoders may actually implement the same partition. For example: if $\mathcal{X} = \{1, 2, 3\}$ and $M = 2$, consider the two encoders:

$e_1 : \ e_1(1) = 1, \ e_1(2) = 2, \ e_1(2) = 2$

$e_2 : \ e_2(1) = 2, \ e_2(2) = 1, \ e_2(2) = 1$

It is easy to see that they have the same functionality. In our definition we treat these encoders as the same encoder, otherwise, the same expert will be taken into account several times. The number of times depends on the specific partition, so we will get an unbalanced set of experts.

1) Definition of the encoders using the partition matrix: To define an encoder uniquely, and to get bounds on the cardinality of the general set of encoders, let us define the partition matrix:

$$PM_{j,l} = \begin{cases} 1 & \text{if } x_j \text{ and } x_l \text{ belongs to the same subset.} \\ 0 & \text{else.} \end{cases} \tag{3}$$

where $j, l \in \{1, 2, \ldots, |\mathcal{X}|\}$, are the indexes of the alphabet letters, $x_j, x_l \in \mathcal{X}$, given that we ordered the alphabet in some arbitrary order. Using properties of this matrix, we can derive bounds on the number of encoders:

$$2^{|\mathcal{X}|-1} \leq Number\ of\ PM's \leq 2^{|\mathcal{X}| \log M} \tag{4}$$

so the number of encoders is exponential in $|\mathcal{X}|$. Therefore, using the general set of encoders is a challenge from a computational complexity point of view.

2) Definition of the decoders: We limit our discussion to decoders which satisfy:

$$d(z_i, y_i) = \hat{x}_i, \ where\ e(\hat{x}_i) = z_i \tag{5}$$

From the above definition, we see that every encoder defines a set of possible decoders. This set consists of all combinations of choices of $\hat{x}$ from the set z, for different pairs (z, y). Using the Hamming distortion measure, it is easy to see that there is no point to choose $\hat{x}_i$ outside the subset m_{z_i}, hence this set of decoders is sufficient. For other distortion measures, the results can be generalized straightforwardly to the set of all possible decoders.

3) The set of scalar source codes: We define $\mathcal{F}^{WZ}(M)$ as the set of all scalar WZ source codes with rate $R = \log M$, i.e. all the pairs of scalar encoder and one of its possible decoders, as defined in (5).

Remark. In contrast to our case, when there is a known joint distribution $P(x_i, y_i)$, then given the encoder and y_i, the best strategy for minimizing the Hamming distortion is, of course, maximum likelihood, i.e., choose the most probable x from the subset m_{z_i}, given y_i.

$$\hat{x} = \arg \max_{x \in m_{z_i}} P_{X|Y}(x|y_i) \tag{6}$$

However, in our case, $P_{X|Y}(x|y)$ is unknown or non-existent since $P(x)$, the source statistics, is unknown or non-existent. Therefore, knowing the encoder is not sufficient for determining the best decoder.

B. An on-line WZ coding scheme

In this part, we describe an on-line adaptive scheme for the WZ case based on the results of [3]. For any source sequence x^n, the distortion $\Delta^n_{(e,d)}(x^n)$ of a source code (e, d) is defined by:

$$\Delta^n_{(e,d)}(x^n) = \sum_{i=1}^{n} \Delta_i(x_i) \tag{7}$$

In the case of scalar source code we get $\Delta^n_{(e,d)}(x^n) = \sum_{i=1}^{n} \sum_{y \in \mathcal{X}} P_{Y|X}(y|x_i)\rho(x_i, d(e(x_i), y))$. Given any finite set of scalar source codes (which we call experts) with rate R and zero delay, this scheme (which has the same rate R and zero delay) achieves asymptotically the distortion $\Delta^n_{(e,d)}(x^n)$ of the best source code in the set, for all source sequences x^n. To be more specific, from [3], we know that for any bounded distortion measure ($d(x, \hat{x}) < B, \forall x, \hat{x} \in \mathcal{X}$ for some positive real number B), the following result holds:

Theorem [3]. 1: Let $\mathcal{A}$ be a finite subset of $\mathcal{F}^{WZ}(M)$. Then there exists a sequential source code $(\tilde{e},\tilde{d})$ such that for all $x^n \in \mathcal{X}^n$:

$$\mathbb{E}\{\tfrac{1}{n}[\Delta^n_{(\tilde{e},\tilde{d})}(x^n) - \min_{(e',d')\in\mathcal{A}} \Delta^n_{(e',d')}(x^n)]\} \leq \frac{3B}{(2R)^{\frac{1}{3}}}[\log|\mathcal{A}|]^{\frac{2}{3}} n^{-\frac{1}{3}} \tag{8}$$

where the expectation is taken w.r.t. to a certain randomization of the algorithm which will be described below. For the Hamming distortion measure, we have $B = 1$. The scheme works as follows: Assume that we have some reference set $\mathcal{A}$ of WZ scalar source codes. We divide the time axis, $i = 1, 2, \ldots, n$, into $K = n/l$ consecutive non-overlapping blocks (assuming l divides n). The value of l was optimized in [3] to get minimal redundancy and is equal to $2\{\log(|\mathcal{A}|)n/R^2\}^{\frac{1}{3}}$. At the beginning of each block, i.e., at times $t = (k-1)l, k \in \{1, 2, \ldots, K\}$, we randomly choose an expert according to the exponential weighting probability distribution:

$$Pr\{next\ expert = (e,d)\} = \frac{\exp\{-\eta\Delta^t_{(e,d)}(x^t)\}}{\sum_{(e',d')\in\mathcal{A}} \exp\{-\eta\Delta^t_{(e',d')}(x^t)\}} \tag{9}$$

where $\eta > 0$ is a parameter that was optimized in [3] to get minimal redundancy and is equal to $\{8\log(|\mathcal{A}|)/lB^2n\}^{\frac{1}{2}}$. After choosing the expert (e,d), the encoder dedicates the first $\lceil \log|\mathcal{A}|/R \rceil$ channel symbols, at the begining of the k-th block, to inform the decoder the identity of d. At the remainder of the block, the encoder produces the channel symbols $z_i = e(x_i)$. At the same time, at the decoder side, at the begining of the block, at times $i = (k-1)l+1, \ldots, \lceil \log|\mathcal{A}|/R \rceil$, the decoder outputs arbitrary sequence of x_i's. At the rest of the block, knowing d, it reproduces $\hat{x}_i = d(z_i, y_i)$.

Remark. Throughout this paper we assume that n is known in advance. Generalizing the scheme to the case where the length is unknown is straightforward, as explained in [3].

III. Efficient implementation for sets of scalar source codes

In this section, we present an efficient implementation of the scheme described, for sets of scalar source codes. Each one of these sets of source codes consists of all the pairs (e,d) where e is one of the encoders in some small set of encoders, and d is one of its possible decoders, as defined in section A. By "small set" we mean that the random choice of the encoder can by done directly (as will be explained below). This definition depends of course on the computational resources we allocate. Remember that given a specific encoder, the decoder, for each (z,y), chooses some x from the subset of source letters m_z. Thus, for each pair (z,y) there are $|m_z|$ possible $\hat{x}$'s. Hence, given an encoder, the number of possible decoders is:

$$\prod_{y\in\mathcal{X}}\prod_{z=1}^{M}|m_z| = (|m_1||m_2|\ldots|m_M|)^{|\mathcal{X}|} \geq 2^{|\mathcal{X}|} \tag{10}$$

where $|m_z|$ is the cardinality of the subset of letters m_z. The lower bound is derived from the fact that in the lossy encoding case $M < |\mathcal{X}|$ so the product above is at least 2. Thus, given a set of encoders, the number of possible WZ source codes is $\geq |E|2^{|\mathcal{X}|}$ where $|E|$ is the number of encoders. Given a set of experts $\mathcal{A}$, we follow the scheme of the previous subsection. We divide the time axis, $i = 1, 2, \ldots, n$, into $K = n/l$ consecutive non-overlapping blocks. We randomly choose the next expert at the beginning of each block according to the exponential weighting probability distribution. The distortion of an expert (e,d) at time t is given by:

$$\begin{aligned}\Delta^t_{(e,d)}(x^t) &= \textstyle\sum_{i=1}^t \Delta(x_i) \\ &= \textstyle\sum_{i=1}^t \sum_y P_{Y|X}(y|x_i)\rho(x_i,\hat{x}(x_i,y)) \\ &= \textstyle\sum_{i=1}^t \sum_y P_{Y|X}(y|x_i)I_{(x_i,y)\in A} \\ &= \textstyle\sum_{x,y\in\mathcal{X}} n_t(x)P_{Y|X}(y|x_i)I_{(x_i,y)\in A}\end{aligned} \tag{11}$$

where A is the set of all pairs (x,y) which contribute to the distortion, i.e., $d(e(x),y) \neq x$, I_B is the indicator function for an event B, and $n_t(x)$ is the number of times x appeared in x^t. For a more convenient form of (9), we multiply the numerator and denominator with $\exp\{\eta\sum_{x,y\in\mathcal{X}} n_t(x)P_{Y|X}(y|x)\}$ and we get:

$$Pr\{next\ expert = (e,d)\} = \frac{\lambda_{(e,d),t}}{\sum_{(e',d')\in\mathcal{A}}\lambda_{(e',d'),t}} \tag{12}$$

where:

$$\lambda_{(e,d),t} = \exp\{\eta\sum_{x,y\in\mathcal{X}} n_t(x)P_{Y|X}(x,y)I_{(x,y)\in\bar{A}}\} \tag{13}$$

where $\bar{A}$ is the complementary set of A, i.e. all pairs (x,y) such that $d(e(x),y) = x$. Given a set of experts, the random choice of an expert at the beginning of each block is done in two steps. First, we choose an encoder randomly according to:

$$Pr\{next\ encoder = e\} = \frac{F_e}{\sum_{e'\in E} F_{e'}} \tag{14}$$

where E is the set of encoders, and:

$$F_e = \sum_{(e,d)\in\mathcal{A}_e} \lambda_{(e,d),t} \tag{15}$$

is the sum of the exponential weights of all experts in $\mathcal{A}_e$, where $\mathcal{A}_e$ is the subset of all experts which use the encoder e. F_e can be calculated efficiently in the following way: For each pair (x,y) calculate $\lambda_{x,y,t}$ where:

$$\lambda_{x,y,t} = \exp\{\eta n_t(x)P_{Y|X}(y|x)\} \tag{16}$$

and then for each (z,y), calculate the sum $\sum_{x:e(x)=z}\lambda_{x,y,t}$ where $e(x)$ is the encoding of x. The product of all these sums is F_e:

$$F_e = \prod_{z=1}^{M}\prod_{y\in\mathcal{X}}\left(\sum_{x:e(x)=z}\lambda_{x,y,t}\right) \tag{17}$$

The proof of this will be given in the full version of this paper [10]. In the second step, we choose the decoder randomly according to:

$$Pr\{decoder = d \mid encoder = e\} = \frac{\lambda_{(e,d),t}}{F_e} \tag{18}$$

The random choice of the decoder can be implemented efficiently in the following way: For each pair (z, y), choose the decoder output $d(z, y)$ randomly, according to the probability distribution:

$$Pr\{d(z,y) = x\} = \frac{\lambda_{x,y,t}}{\sum_{x':e(x')=z} \lambda_{x',y,t}} \tag{19}$$

where $x \in \{x : e(x) = z\}$.

We demonstrated an efficient random choice of a pair (e, d). The proof that this random choice is actually implementing (9) will be given in [10]. Below is a formal description of the online algorithm:

1) Calculate l, the optimal length of a data block, and let $K = n/l$.
2) Initialize k to 0, and all the weights $\lambda_{x,y,0}$ to 1.
3) At the beginning of each of the next data blocks, update the weights in the following way:
   $\lambda_{x,y,t_k} = \lambda_{x,y,t_{k-1}} \exp(\eta \sum_{i=(k-1)l+1}^{kl} I_{x_i=x} P_{Y|X}(x, y))$
   $t_k = kl + 1,\ 1 \le k \le K - 1$
4) For each (e, z, y), calculate the sums:
   $\sum_{x:e(x)=z} \lambda_{x,y,t_k}$
5) Calculate F_e, for each $e \in E$, according to (17).
6) Choose an encoder e_k randomly according to (14).
7) For each pair (z, y), choose the decoder function d_k randomly according to (19).
8) Use the first $\lceil \log(N)/R \rceil$ channel symbols at the beginning of the kth block to inform the decoder the identity of d_k, chosen in the previous step, where N is the number of experts.
9) Encode the next block using the chosen expert e_k:
   $z_i = e_k(x_i),\ kl + \log(N)/R + 1 \le i \le (k + 1)l - 1$
10) If $k < K$, increment k and go to 3.

The total complexity of the algorithm is $O(n/l \cdot |\mathcal{X}|^2) + O(n/l \cdot |E||\mathcal{X}|M) + O(n)$. The complexity depends on $|E|$, which thus should be small as was mentioned above.

A. Large set of encoders with structure

As was shown, we can choose a pair (e, d) randomly, in two steps. In the first step, we choose the encoder according to (14). In the second step, we choose randomly one of its possible decoders according to (18). In the previous part, we assumed that the set of encoders is small, so we can implement (14) directly, i.e., calculate F_e for each encoder separately. In this part, we use a large structured set of encoders. Using the structure, we can efficiently implement (14). We assume that the input alphabet $\mathcal{X}$ is ordered. We enumerate the source symbols according to that order. By $Num(x), 1 \le Num(x) \le |\mathcal{X}|$, we denote the serial number of the symbol x.

1) Definition of the set of encoders: The Input Alphabet Axis (IAA) is defined as the $|\mathcal{X}|$-sized vector $(1, 2, \ldots, |\mathcal{X}|)$. A division of the IAA is given by the $(M-1)$-sized increasing sequence $\mathbf{r} = (z_1, \ldots, z_{M-1}), z_i \in \{1, 2, \ldots |\mathcal{X}|\}$. Each division r represents a specific encoder in the following way:

$$e(x) = i : z_{i-1} \le Num(x) < z_i, \quad i \in \{1, \ldots, M\} \tag{20}$$

We define E as the set of all such encoders. The cardinality of the set of encoders is $\binom{|\mathcal{X}|}{M-1}$.

2) Graphical representation of the set of encoders: The random choice of the encoders can be done efficiently using an a-cyclic directed graph. We denote:
$\mathcal{V}$ - The set of all vertices:
$\{1, 2, \ldots, M-1\} \times \{1, 2, \ldots, |\mathcal{X}|\} \cup (0, 0) \cup (M, |\mathcal{X}| + 1)$
$\mathcal{E}$ - The set of all edges:
$\{((z, j-1), (\hat{z}, j)) : z, \hat{z} \in \{0, 1, 2, \ldots, M+1\}, \hat{z} > z, j \in \{1, 2, \ldots, |\mathcal{X}| + 1\}\}$
s - The starting point in the bottom left, i.e. $(0, 0)$
u - The end point in the top right, i.e. $(M + 1, |\mathcal{X}| + 1)$
$\mathcal{E}_z$ - The set of all edges starting from vertex z.
The graph is described in Fig.1. The horizontal axis represents the ordered input alphabet. The vertical axis represents the $M-1$ choices needed for dividing the IAA into M segments. A path composed of the edges $\{(0, 0), (z_1, 1) \ldots, (z_{M-1}, M-1), (|\mathcal{X}|+1, M)\}$ represents $M-1$ consecutive choices of $M-1$ x's $(z_1, \ldots, z_{M-1})$ which divide the IAA into M segments, creating M subsets of the input alphabet. Each edge on a path represents one choice, the choice of the next point on the vertical axis, which defines the next segment. An edge $((z, j-1), (\hat{z}, j))$ matches to the segment $[z, \hat{z})$ on the vertical axis, thus equivalent to the subset $\{x : z \le x < \hat{z}\}$. There are $O(M|\mathcal{X}|^2)$ edges. For each edge $a \in \mathcal{E}$ and time t we assign

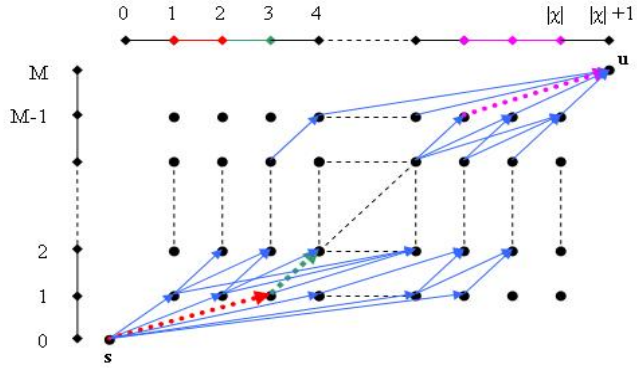

Fig. 1. The graph representing all possible partitions of the input alphabet into M subsets given the alphabet is ordered in some specific order. For example, the left dashed arrow defines the subset $\{1, 2\}$, the middle and right dashed arrows define the subsets $\{3\}$ and $\{|\mathcal{X}|-2, |\mathcal{X}|-1, |\mathcal{X}|\}$ respectively.

a weight $\delta_{a,t}$:

$$\delta_{a,t} = \prod_{y \in \mathcal{X}} \sum_{x \in [z, \hat{z})} \lambda_{x,y,t}, \quad a = ((z, j-1), (\hat{z}, j)) \tag{21}$$

where $\lambda_{x,y,t}$ is given by (16). It can be seen from (21) that a weight $\delta_{a,t}$ depends only on the vertical coordinates of the edge a, thus we can denote it as $\delta_{(z,\hat{z}),t}$. The cumulative weight of a path $r = \{(0, 0), (z_1, 1) \ldots, (z_{M-1}, M-1), (|\mathcal{X}|+1, M)\}$ at time t is defined as the product of its edges' weights:

$$\Lambda_{r,t} = \prod_{a \in r} \delta_{a,t} \tag{22}$$

It will be shown in [10] that $\Lambda_{r,t} = F_e$. Following the WPA, also used in [5] we define:

$$G_t(z) = \sum_{r\in\mathcal{R}_z} \prod_{a\in r} \delta_{a,t} \tag{23}$$

where z is a vertex on the graph, $\mathcal{R}_z$ is the set of all paths from z to u and a is an edge on the path r.
It is easy to see that:

$$G_t(s) = \sum_{r\in\mathcal{R}} \Lambda_{r,t} = \sum_{e\in E} F_e \tag{24}$$

The function $G_t(z)$ can be computed recursively:

$$G_t(u) = 1, \quad G_t(z) = \textstyle\sum_{\hat{z}:(z,\hat{z})\in\mathcal{E}} \delta_{(z,\hat{z}),t} G_t(\hat{z}) \tag{25}$$

Because each edge is taken exactly once, calculating $G_t(z)$ for all z's requires $O(|\mathcal{E}|)$ computations given the weights $\delta_{a,t}$. The function $G_t(z)$ offers an efficient way to choose an encoder randomly according to probability distribution in (14). We define for each $\hat{z} \in \mathcal{E}_z$:

$$P_t(\hat{z}|z) = \delta_{(z,\hat{z}),t} G_t(\hat{z})/G_t(z) \tag{26}$$

It is easy to see that $P_t(\hat{z}|z)$ is indeed a probability distribution, i.e., $\sum_{\hat{z}:(z,\hat{z})\in\mathcal{E}_z} P_t(\hat{z}|z) = 1$. We also have:

$$\prod_{m=1}^{M} P_t(z_{m,r}|z_{m-1,r}) = \prod_{m=1}^{M} \delta_{(z_{m-1,r},z_{m,r}),t} \frac{G_t(z_{m,r})}{G_t(z_{m-1,r})}$$

$$= \frac{G_t(u)}{G_t(s)} \cdot \prod_{m=1}^{M} \delta_(z_{m-1,r}, z_{m,r}) = Pr\{next\ encoder = r\}$$

and we get exactly the probability in (14).
Therefore, the encoder can be chosen randomly in the following sequential manner: Starting from $z_0 = s$, in each step $m = 1, 2, \ldots, M-1$ choose the next vertex $z_m \in \mathcal{E}_{z_{m-1}}$ with probability $P_t(z_m|z_{m-1})$. The procedure stops when $z_m = u$.

Formal description of the on-line algorithm: Using the set of encoders described above, we now have the following algorithm:

1) Calculate l, the optimal length of a data block, and let $K = n/l$.
2) Initialize k to 0, and all the weights $\lambda_{x,y,0}$ to 1.
3) Build the encoders graph as described in this section.
4) Initialize all the weights $\delta_{a,0}$ to 1.
5) At the beginning of each of the next data blocks, i.e. at times $t_k = kl + 1, k = \{1, \ldots, K\}$ update the weights in the following way:
   $\lambda_{x,y,t_k} = \lambda_{x,y,t_{k-1}} \exp(\eta \sum_{i=(k-1)l+1}^{kl} I_{x_i=x} P_{Y|X}(x,y))$
6) At the beginning of each of the next data blocks, calculate $\delta_{z,\hat{z},t_k}$ for each pair $(z,\hat{z})$ according to (21).
7) Update the weights of all edges to the new $\delta_{(z,\hat{z}),t_k}$'s.
8) Calculate $G_{t_k}(z)$ recursively, for all z, according to (25).
9) Choose the encoder e_k randomly as described above, using (26).
10) For each pair (z,y), choose the decoder function d_k randomly according to (18).
11) Use the first $\lceil \log(N)/R \rceil$ channel symbols at the beginning of the kth block to inform the decoder the identity of d_k, chosen in the previous step, where N is the number of experts.
12) Encode the next block, using the chosen expert e_k:
    $z_i = e_k(x_i),\ kl + \log(N)/R + 1 \le i \le (k+1)l - 1$
13) If $k < K$, increment k and go to 3.

The total complexity of the algorithm is $O(n/l \cdot |\mathcal{X}|^3) + O(n/l \cdot M|\mathcal{X}|^2) + O(n/l|\mathcal{X}|^2) + O(n)$.

B. General distortion measures

Let $\rho(x,\hat{x})$ be some bounded distortion measure ($d(x,\hat{x}) < B, \forall x,\hat{x} \in \mathcal{X}$ for some positive real number B). Given an encoder e, we define:

$$\lambda_{x,y,e,t} = \exp\left\{-\eta \textstyle\sum_{\{x':x'\neq x, e(x)=e(x')\}} n_t(x') P_{Y|X}(y|x') \rho(x,x')\right\} \tag{27}$$

It is easy to see that given some possible decoder d, the distortion of the pair $r = (e,d)$ is:

$$\lambda_{r,t} = \prod_{y\in\mathcal{X}} \prod_{m=1}^{M} \lambda_{d(z,y),y,e,t} \tag{28}$$

using the generalized λ's we defined, we can continue exactly as in the Hamming case. It will be shown in [10] that the cost of using the generalized λ is increased complexity by a factor of $|\mathcal{X}|$.

ACKNOWLEDGMENT

This research is supported by the Israeli Science Foundation (ISF), grant no. 208/08.

REFERENCES

[1] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pt. I, pp. 379–423, 1948; pt. II, pp. 623–656, 1948.
[2] T. Linder and G. Lugosi, "A zero-delay sequential scheme for lossy coding of individual sequences," *IEEE Trans. Inform. Theory*, vol. 46, pp. 2533–2538, Sept. 2001.
[3] T. Weissman and N. Merhav, "On limited-delay lossy coding and filtering of individual sequences," *IEEE Trans. Inform. Theory*, vol. 48, pp. 721–733, Mar. 2002.
[4] A. György, T. Linder and G. Lugosi, "Efficient adaptive algorithms and minimax bounds for zero-delay lossy source coding," *IEEE Trans. Signal. Proc.*, vol. 52, pp. 2337–2347, Aug. 2004.
[5] A. György, T. Linder and G. Lugosi, "Tracking the best quantizer," *IEEE Trans. Inform. Theory*, vol. 54, pp. 1604–1625, April. 2008.
[6] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Trans. Inform. Theory*, vol. IT-22, pp. 1–10, Jan. 1976.
[7] D. Muresan and M Effros, "Quantization as histogram segmentation: Optimal scalar quantizer design in network systems," *IEEE Trans. Inform. Theory*, vol. 54, pp. 344–366, Jan. 2008.
[8] J. Kusuma, L. Doherty and K. Ramchandran, "Distributed compression for sensor networks," *ICIP conf.*, vol. 1, pp. 82–85, 2001.
[9] N. Merhav and J. Ziv, "On the WynerZiv Problem for Individual Sequences," *IEEE Trans. Inform. Theory*, vol. 52, pp. 867-873, Mar. 2006.
[10] A. Reani and N. Merhav, "Efficient on-line schemes for encoding individual sequences with side information at the decoder," in preperation.