Coding for a binary independent piecewise-identically-distributed source

Correspondence

Coding for a Binary Independent Piecewise-Identically-Distributed Source

Frans M. J. Willems, *Member, IEEE*

Abstract— **Two weighting procedures are presented for compaction of output sequences generated by binary independent sources whose unknown parameter may occasionally change. The resulting codes need no knowledge of the sequence length T, i.e., they are strongly sequential, and also the number of parameter changes is unrestricted. The additional-transition redundancy of the first method was shown to achieve the Merhav lower bound, i.e., $\log T$ bits per transition. For the second method we could prove that additional-transition redundancy is not more than $\frac{3}{2}\log T$ bits per transition, which is more than the Merhav bound; however, the storage and computational complexity of this method are also more interesting than those of the first method. Simulations show that the difference in redundancy performance between the two methods is negligible.**

Index Terms— **Universal noiseless source coding, independent piecewise-identically-distributed source, nonstationary source, transition pattern, weighting, arithmetic coding, redundancy bounds.**

I. Introduction

The Elias Algorithm (described in, e.g., Jelinek [1]) produces for any coding distribution $P_c(x_1^T)$ over all binary sequences $x_1^T = x_1x_2\cdots x_T \in \{0,1\}^T$, a binary prefix code with codeword lengths $L(x_1^T)$ such that[1]

$$L(x_1^T) < \log \frac{1}{P_c(x_1^T)} + 2 \quad \text{for all } \textit{possible } x_1^T. \tag{1}$$

Possible sequences are sequences that can actually occur, i.e., sequences x_1^T with actual probability $P_a(x_1^T) > 0$. It is required that the coding distribution $P_c(\cdot)$ satisfies

$$\begin{aligned} &P_c(\phi) = 1,\\ &P_c(x_1^{t-1}) = P_c(x_1^{t-1}, X_t = 0) + P_c(x_1^{t-1}, X_t = 1),\\ &\qquad \text{for all } x_1^{t-1} \in \{0,1\}^{t-1}, t = 1,\cdots,T, \text{ and}\\ &P_c(x_1^T) > 0 \quad \text{for all } \textit{possible } x_1^T \in \{0,1\}^T. \end{aligned} \tag{2}$$

Note that ϕ stands for the empty sequence (x_1^0). If the marginals $P_c(x_1^t), t = 1,\cdots,T$ are sequentially available the *arithmetic* code can be implemented sequentially. After having accepted a *coding redundancy* of at most 2 bits, we are left with the problem of finding good coding distributions $P_c(\cdot)$ for the cases we are interested in.

For binary independent and identically distributed (i.i.d.) sources with unknown parameter θ (i.e., the probability of generating a 1), we can assign the block probability $P_c(x_1^T) = P_e(a,b)$ to a sequence x_1^T containing a zeros and b ones where

$$P_e(a,b) \triangleq \int_{0,1} \frac{1}{\pi\sqrt{(1-\theta)\theta}}(1-\theta)^a\theta^b \, d\theta. \tag{3}$$

This distribution allows sequential updating, i.e., $P_e(0,0) = 1$, and for $a \geq 0$ and $b \geq 0$

$$P_e(a+1,b) = \frac{a+1/2}{a+b+1} \cdot P_e(a,b)$$

and

$$P_e(a,b+1) = \frac{b+1/2}{a+b+1} \cdot P_e(a,b). \tag{4}$$

It was suggested by Krichevsky and Trofimov [2] and satisfies (see [10]), for $a + b \geq 1$, the inequality

$$P_e(a,b) \geq \frac{1}{2} \cdot \frac{1}{\sqrt{a+b}} \left(\frac{a}{a+b}\right)^a \left(\frac{b}{a+b}\right)^b. \tag{5}$$

Using this inequality we can easily upper-bound the *parameter redundancy* resulting from this estimator. If x_1^T is a (nonempty) sequence containing a zeros and b ones, with *actual* probability $P_a(x_1^T|\theta)$, we obtain for the parameter redundancy relative to the actual source that

$$\begin{aligned} \log \frac{P_a(x_1^T|\theta)}{P_c(x_1^T)} &= \log \frac{(1-\theta)^a\theta^b}{P_e(a,b)}\\ &\leq \log \frac{\left(\frac{a}{a+b}\right)^a \left(\frac{b}{a+b}\right)^b}{\frac{1}{2} \cdot \frac{1}{\sqrt{a+b}} \left(\frac{a}{a+b}\right)^a \left(\frac{b}{a+b}\right)^b}\\ &= \frac{1}{2}\log T + 1, \quad \text{for all } \theta \in [0,1]. \end{aligned} \tag{6}$$

In other words, the parameter redundancy is uniformly bounded. The inequality in (6) follows from applying the lower bound in (5) for the denominator and observing that the maximum value of the enumerator is achieved for $\theta = b/(a+b)$. In a more general setting the memoryless source is not identically distributed. We assume in this correspondence that the source parameter θ changes occasionally.

Definition 1: An *independent piecewise-identically-distributed (i.p.i.d.)* binary source generates a sequence $x_1^T \in \{0,1\}^T$ of independent symbols. During this generation process, the source parameter θ makes, say C, transitions and assumes $C+1$ values. The *transition pattern* $\mathcal{T} = (t_1, t_2, \cdots, t_C)$ for sequence x_1^T indicates that the symbols $x_{t_c}, x_{t_c+1}, \cdots, x_{t_{c+1}-1}$ are generated according to parameter θ_c, for $c = 0, C$. We assume here that $t_0 \triangleq 1$ and $t_{C+1} \triangleq T+1$. Note that $t_{c+1} > t_c$ for $c = 0, C$ and, consequently, $C \leq T-1$. We can now write for the actual probability

$$P_a(x_1^T = x_1^T|\mathcal{T},\Theta) = \prod_{t=1,T} P_a(X_t = x_t)$$

where

$$\begin{aligned} P_a(X_t = 1) &= 1 - P_a(X_t = 0) = \theta_c,\\ &\quad \text{if } t_c \leq t < t_{c+1} \text{ for } c = 0,1,\cdots,C. \end{aligned} \tag{7}$$

The vector consisting of the parameters $\theta_0, \theta_1, \cdots, \theta_c$ is denoted by Θ and called the *parameter vector*. We write $C(\mathcal{T})$ for the number of transitions C in pattern $\mathcal{T}$. □

Example: Our source generates 1000 binary digits. The starting value of the parameter of our example source is 0.9. This parameter changes into 0.0 at $t = 301$. At $t = 401$ the parameter value becomes 0.5. Finally, at (before) $t = 601$ there is a transition again and

Manuscript received March 2, 1995; revised April 12, 1996. The material in this correspondence was presented at the IEEE International Symposium on Information Theory, Whistler, BC, Canada, September 17–22, 1995.

The author is with the Electrical Engineering Department, Eindhoven University of Technology, P. O. Box 513, 5600 MB Eindhoven, The Netherlands.

Publisher Item Identifier S 0018-9448(96)06890-3.

[1] We use the notation of [10]. It is assumed that the base of the $\log(\cdot)$ is 2. Codeword lengths and information quantities are both expressed in bits.

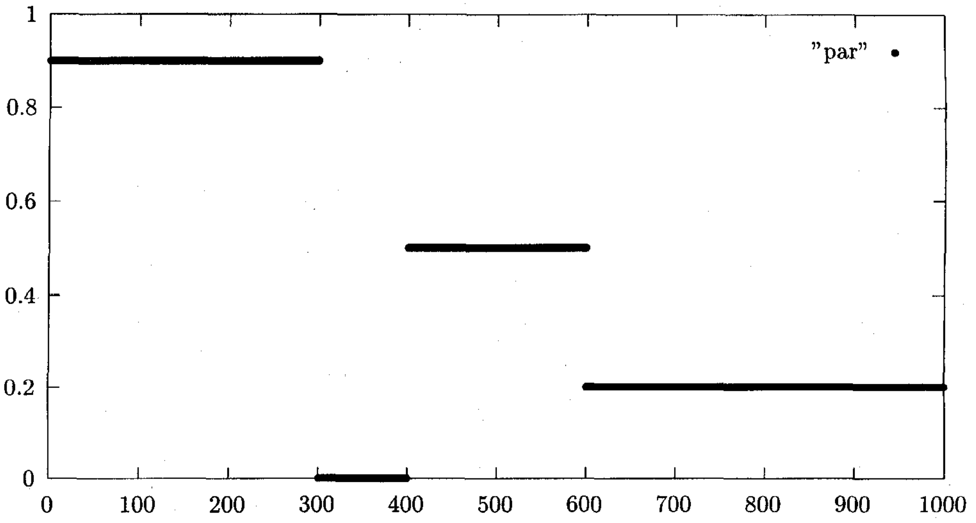


Fig. 1. Value of the parameter θ as a function of t for $t = 1, 2, \cdots, 1000$.

the new value becomes 0.2. The parameter now remains constant until the end of the sequence. This parameter behavior is plotted in Fig. 1. The number of transitions is three, the transition pattern is $(301, 401, 601)$, the parameter vector $(0.9, 0.0, 0.5, 0.2)$.

If we know the actual transition pattern $\mathcal{T}$, we can partition the sequence x_1^T into $C(\mathcal{T}) + 1$ memoryless subsequences and use

$$P_c(x_1^T|\mathcal{T}) = \prod_{c=0,C(\mathcal{T})} P_e(a(x_{t_c}^{t_{c+1}-1}), b(x_{t_c}^{t_{c+1}-1})) \tag{8}$$

as a coding distribution, where $a(x_i^j)$ (resp., $b(x_i^j)$) is the number of zeros (resp., ones) that occur in $x_i^j = x_i x_{i+1} \cdots x_j$. Again this coding distribution allows sequential updating. For any sequence $x_1^T \in \{0,1\}^T$, using (6) and the convexity of the $\log(\cdot)$, the parameter redundancy with respect to the actual transition pattern $\mathcal{T}$ and the actual parameter vector Θ, can be upper bounded as

$$\log \frac{P_a(x_1^T|\mathcal{T},\Theta)}{P_c(x_1^T|\mathcal{T})} \le \frac{C(\mathcal{T})+1}{2} \log \frac{T}{C(\mathcal{T})+1} + C(\mathcal{T}) + 1, \quad \text{for all } \Theta \in [0,1]^{C(\mathcal{T})+1}. \tag{9}$$

Note that we do not need a function $\gamma(\cdot)$ as in [10] here since always $t_{c+1} > t_c$ for $c = 0, C(\mathcal{T})$.

If the transition pattern is unknown we can *weight* the coding distributions corresponding to all transition patterns $\mathcal{T}$ of sequences of length T and obtain the weighted coding distribution

$$P_c(x_1^T) = \sum_{\mathcal{T}} P_{\text{tr}}(\mathcal{T}) P_c(x_1^T|\mathcal{T}).$$

Here $P_{\text{tr}}(\mathcal{T})$ is the *a priori* probability that is assigned to transition pattern $\mathcal{T}$. Of course

$$\sum_{\mathcal{T}} P_{\text{tr}}(\mathcal{T}) = 1.$$

For any sequence $x_1^T \in \{0,1\}^T$ the *transition redundancy* with respect to pattern $\mathcal{T}$ can be upper-bounded as

$$\log \frac{P_c(x_1^T|\mathcal{T})}{P_c(x_1^T)} \le \log \frac{1}{P_{\text{tr}}(\mathcal{T})} \quad \text{for all } \mathcal{T} \tag{10}$$

simply by noting that $P_c(x_1^T) \ge P_{\text{tr}}(\mathcal{T}) P_c(x_1^T|\mathcal{T})$.

The *total cumulative redundancy* relative to the actual pattern $\mathcal{T}$ and actual vector Θ is equal to the sum of the (cumulative) transition, parameter and coding redundancies. Using (1), (9), and (10) we can upper-bound this total redundancy for any sequence x_1^T in the following way:

$$\begin{aligned} L(x_1^T) &- \log \frac{1}{P_a(x_1^T|\mathcal{T},\Theta)} \\ &= \log \frac{P_c(x_1^T|\mathcal{T})}{P_c(x_1^T)} + \log \frac{P_a(x_1^T|\mathcal{T},\Theta)}{P_c(x_1^T|\mathcal{T})} + L(x_1^T) - \log \frac{1}{P_c(x_1^T)} \\ &< \log \frac{1}{P_{\text{tr}}(\mathcal{T})} + \frac{C(\mathcal{T})+1}{2} \log \frac{T}{C(\mathcal{T})+1} + C(\mathcal{T}) + 3. \end{aligned} \tag{11}$$

This holds for all transition patterns $\mathcal{T}$ of x_1^T and parameters $\Theta \in [0,1]^{C(\mathcal{T})+1}$. Rewriting this bound, and taking the minimum over all transition patterns $\mathcal{T}$ of x_1^T and all parameter vectors Θ, we obtain for all $x_1^T \in \{0,1\}^T$ (not necessarily being generated by the source with transition pattern $\mathcal{T}$ and parameter vector Θ) that

$$\begin{aligned} L(x_1^T) < \min_{\mathcal{T},\Theta} \Bigg\{ &\log \frac{1}{P_a(x_1^T|\mathcal{T},\Theta)} + \log \frac{1}{P_{\text{tr}}(\mathcal{T})} \\ &+ \frac{C(\mathcal{T})+1}{2} \log \frac{T}{C(\mathcal{T})+1} + C(\mathcal{T}) + 3 \Bigg\}. \end{aligned} \tag{12}$$

From this, we may conclude that weighting methods try to minimize the *total description length* of a sequence. In the following sections we will describe certain *a priori* distributions over the transition patterns, and investigate the coding methods that are based on these distributions.

There is a vast amount of publications in the image, audio, and speech processing literature on segmentation problems. Although at first sight the problem treated here is similar to all these segmentation problems, the approach taken here is entirely different. Our objective is not to determine the boundary points of the (stationary) segments but simply to compress the entire sequence. It is a bit surprising to realize that also in the image, audio, and speech processing literature compression is often the primary goal. However, it seems advantageous to replace this primary objective by a secondary objective, namely, to determine segments of constant statistics.

II. A Quadratical-Complexity Coding Method

We will first define a coding distribution over source sequences and transition patterns that have infinite length in principle. We will see later that the storage complexity resulting from this distribution increases quadratically with the sequence length T. Therefore, we refer to this method as the *quadratical-complexity coding method.*

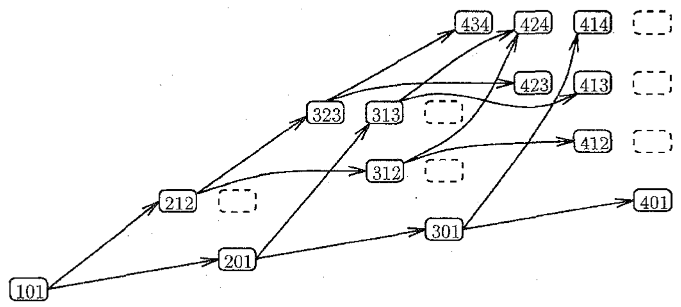


Fig. 2. Quadratical transition diagram for $t = 1, 2, 3, 4$. State (t, c, t_c) is denoted as tct_c.

We can define the distribution after assuming that the source follows a path in a quadratical *transition diagram* (see Fig. 2).

Definition 2: The source starts in *state* $(1, 0, 1)$ of the quadratical transition diagram. It generates the first output x_1 according to parameter θ_0. When the source produces output x_t, it is in a state (t, c, t_c), which indicates that the number of parameter*changes* encountered so far is c, that the current parameter is θ_c, and that the last time index *before* which a change occurred is t_c. To obtain Krichevsky–Trofimov estimators we assign, in state (t, c, t_c), the following probabilities to the values 0 and 1 that can be generated:

$$P_c(X_t = 0|(t, c, t_c)) = \frac{a(x_{t_c}^{t-1}) + 1/2}{t - t_c + 1}$$
$$P_c(X_t = 1|(t, c, t_c)) = \frac{b(x_{t_c}^{t-1}) + 1/2}{t - t_c + 1}. \quad (13)$$

Again, $a(x_i^j)$ (resp., $b(x_i^j)$) is the number of zeros (resp., ones) that occurs in $x_i^j = x_i x_{i+1} \cdots x_j$. After having generated x_t, the parameter can change and the source will go to state $(t+1, c+1, t+1)$. If the parameter remains the same, the next state of the source will be $(t+1, c, t_c)$. We assign the following probabilities to these two alternatives to obtain Krichevsky–Trofimov estimators:

$$P_{\text{tr}}((t+1, c, t_c)|(t, c, t_c)) = \frac{t - c - 1/2}{t}$$
$$P_{\text{tr}}((t+1, c+1, t+1)|(t, c, t_c)) = \frac{c + 1/2}{t}. \quad (14)$$

We should note that state (t, c, t_c) only exists if $0 \le c \le t-1$ and if $c+1 \le t_c \le t$ for $c \ne 0$ and $t_0 = 1$. The source output x_t and the next state $(t+1, p, q)$, where $(p, q) = (c, t_c)$ or $(c+1, t+1)$ are independent of each other given the current state (t, c, t_c). Note that the conditional coding probabilities in (13) do not depend on c while the conditional transition probabilities in (14) do not depend on t_c.□

We can now specify the quadratical-complexity coding distribution over all source sequences.

Definition 3: For the quadratical method, for each sequence $x_1^T \in \{0,1\}^T$ we compute for all $x_1^t \in \{0,1\}^t, t = 1, T$, the probabilities $P_c(x_1^t, (t, c, t_c))$ recursively, i.e.,

$$P_c(x_1^t, (t, p, q)) \triangleq \sum_{r,s} P_c(x_1^{t-1}, (t-1, r, s)) \cdot P_{\text{tr}}((t, p, q)|(t-1, r, s)) P_c(x_t|(t, p, q)) \quad (15)$$

for $t > 1$, and $P_c(x_1, (1, 0, 1)) = 1/2$ for both $x_1 = 0$ and 1. The summation is only over r, s such that $(t-1, r, s)$ is an existing state in the diagram (connected to state (t, p, q)).

The coding probabilities $P_c(x_1^t), x_1^t \in \{0,1\}^t, t = 1, T$, are now defined as follows:

$$P_c(x_1^t) \triangleq \sum_{p,q} P_c(x_1^t, (t, p, q)). \quad (16)$$

The summation is only over p, q such that (t, p, q) is an existing state in the transition diagram. Of course, $P_c(\phi) = 1$. □

It is easy to show that the coding probabilities $P_c(x_1^t), t = 0, T$, satisfy the restrictions (2). After having discussed the implementation of this method it will be clear that this coding distribution is sequentially available.

It is possible to demonstrate that this coding distribution performs a weighting of all coding distributions $P_c(x_1^T|\mathcal{T})$ over all transition patterns $\mathcal{T}$ over T source symbols, i.e.,

$$P_c(x_1^T) = \sum_{p,q} \sum_{\mathcal{T} \to (T,p,q)} P_{\text{tr}}(\mathcal{T}) P_c(x_1^T|\mathcal{T}) \quad (17)$$

where

$$\sum_{\mathcal{T} \to (T,p,q)}$$

means a summation over all $\mathcal{T}$ that lead to state (T, p, q). Furthermore, for a transition pattern $\mathcal{T}$ such that $\mathcal{T} \to (T, p, q)$, i.e.. a pattern $\mathcal{T}$ that leads to (T, p, q)

$$P_{\text{tr}}(\mathcal{T}) = P_e(T - 1 - p, p). \quad (18)$$

Probability $P_c(x_1^T|\mathcal{T})$ is as defined in (8). This result is a direct consequence of the following lemma.

Lemma 1: For $t = 1, T$ and any existing state (t, p, q) in the quadratical transition diagram and all $x_1^t \in \{0,1\}^t$

$$P_c(x_1^t, (t, p, q)) = \sum_{\mathcal{T} \to (t,p,q)} P_{\text{tr}}(\mathcal{T}) P_c(x_1^t|\mathcal{T}) \quad (19)$$

with

$$P_{\text{tr}}(\mathcal{T}) = P_e(t - 1 - p, p) \quad (20)$$

and

$$P_c(x_1^t|\mathcal{T}) = \left(\prod_{c=0,p-1} P_e(a(x_{t_c}^{t_{c+1}-1}), b(x_{t_c}^{t_{c+1}-1})) \right) \cdot P_e(a(x_{t_p}^t), b(x_{t_p}^t)). \quad (21)$$

Proof: We will prove the lemma by induction. First note that the hypothesis holds for $t = 1$. For $t = 1$ only the state $(1, 0, 1)$ exists and $P_{\text{tr}}((1, 0, 1)) = P_e(0, 0) = 1$ and $P_c(x_1|(1, 0, 1)) = 1/2$ for both $x_1 = 0$ and 1. Next we assume that the hypothesis holds for $t - 1$ when $t = 2, T$. For t we then obtain using (15) and the hypothesis that

$$\begin{aligned} P_c(x_1^t, (t, p, q)) &= \sum_{r,s} \sum_{\mathcal{S} \to (t-1,r,s)} P_{\text{tr}}(\mathcal{S}) P_c(x_1^{t-1}|\mathcal{S}) \\ &\quad \cdot P_{\text{tr}}((t, p, q)|(t-1, r, s)) P_c(x_t|(t, p, q)) \\ &= \sum_{r,s} \sum_{\mathcal{S} \to (t-1,r,s)} P_{\text{tr}}(\mathcal{S}, (t, p, q)) \\ &\quad \cdot P_c(x_1^{t-1}, x_t|\mathcal{S}, (t, p, q)) \\ &= \sum_{\mathcal{T} \to (t,p,q)} P_{\text{tr}}(\mathcal{T}) P_c(x_1^T|\mathcal{T}) \end{aligned} \quad (22)$$

where we have used that

$$P_{\text{tr}}(\mathcal{S}) P_{\text{tr}}((t, p, q)|(t-1, r, s)) = P_{\text{tr}}(\mathcal{S}, (t, p, q)) = P_{\text{tr}}(\mathcal{T})$$

and that

$$P_c(x_1^{t-1}|\mathcal{S}) P_c(x_t|(t, p, q)) = P_c(x_1^{t-1}, x_t|\mathcal{S}, (t, p, q)) = P_c(x_1^t|\mathcal{T})$$

for

$$\mathcal{T} = (\mathcal{S}, (t, p, q)) \quad \text{and} \quad \mathcal{S} \to (t-1, r, s).$$

To see that $P_{\text{tr}}(\mathcal{T}) = P_e(t - 1 - p, p)$ as stated in (20), note that from the induction hypothesis we know that $P_e(\mathcal{S}) = P_e(t-2-r, r)$.

Now if $p = r$ then no new parameter change occurred going from $(t-1, r, s)$ to (t, p, q) and $P_{\rm tr}(\mathcal{S})$ is multiplied by $(t-1-r-1/2)/(t-1)$ and we obtain

$$\begin{aligned} P_{\rm tr}(\mathcal{T}) &= P_e(t-2-r, r)\cdot(t-1-r-1/2)/(t-1) \\ &= P_e(t-1-r, r) \\ &= P_e(t-1-p, p). \end{aligned}$$

Alternatively, if $p = r+1$ then a transition occurred. The multiplication factor is now $(r+1/2)/(t-1)$ and we get

$$\begin{aligned} P_{\rm tr}(\mathcal{T}) &= P_e(t-2-r, r)\cdot(r+1/2)/(t-1) \\ &= P_e(t-2-r, r+1) \\ &= P_e(t-1-p, p). \end{aligned}$$

Furthermore, (21) can be understood by noting first that by the induction hypothesis

$$\begin{aligned} P_c(x_1^{t-1}|\mathcal{S}) = &\left(\prod_{c=0,r-1} P_e(a(x_{t_c}^{t_{c+1}-1}), b(x_{t_c}^{t_{c+1}-1}))\right) \\ &\cdot P_e(a(x_{t_r}^{t-1}), b(x_{t_r}^{t-1})). \end{aligned}$$

We assume that $x_t = 0$, but similar remarks apply for $x_t = 1$. If $p = r$ then no new parameter transition occurred going from $(t-1, r, s)$ to (t, p, q) and $P_c(x_1^{t-1}|\mathcal{S})$ is multiplied by

$$(a(x_{t_r}^{t-1}) + 1/2)/(t - t_r + 1)$$

and we obtain

$$\begin{aligned} P_c(x_1^t|\mathcal{T}) = &\left(\prod_{c=0,r-1} P_e(a(x_{t_c}^{t_{c+1}-1}), b(x_{t_c}^{t_{c+1}-1}))\right) \\ &\cdot P_e(a(x_{t_r}^{t-1}), b(x_{t_r}^{t-1})) \\ &\cdot (a(x_{t_r}^{t-1}) + 1/2)/(t-t_r+1) \\ = &\left(\prod_{c=0,r-1} P_e(a(x_{t_c}^{t_{c+1}-1}), b(x_{t_c}^{t_{c+1}-1}))\right) \\ &\cdot P_e(a(x_{t_r}^{t-1}) + 1, b(x_{t_r}^{t-1})) \\ = &\left(\prod_{c=0,p-1} P_e(a(x_{t_c}^{t_{c+1}-1}), b(x_{t_c}^{t_{c+1}-1}))\right) \\ &\cdot P_e(a(x_{t_p}^{t}), b(x_{t_p}^{t})). \end{aligned}$$

On the other hand, when a transition occurred $p = r+1$ and $t_{r+1} = t$. We rewrite

$$P_c(x_1^{t-1}|\mathcal{S}) = \prod_{c=0,r} P_e(a(x_{t_c}^{t_{c+1}-1}), b(x_{t_c}^{t_{c+1}-1})).$$

The multiplication factor $P_c(X_t = 0|(t, r+1, t)) = 1/2$ and we find for

$$\begin{aligned} P_c(x_1^t|\mathcal{T}) = &\left(\prod_{c=0,r} P_e(a(x_{t_c}^{t_{c+1}-1}), b(x_{t_c}^{t_{c+1}-1}))\right) \\ &\cdot P_e(a(x_{t_{r+1}}^{t-1}) + 1, b(x_{t_{r+1}}^{t-1})) \\ = &\left(\prod_{c=0,p-1} P_e(a(x_{t_c}^{t_{c+1}-1}), b(x_{t_c}^{t_{c+1}-1}))\right) \\ &\cdot P_e(a(x_{t_p}^{t}), b(x_{t_p}^{t})). \end{aligned}$$

□

Theorem 1: Let $T \geq 2$. If we use the quadratical-complexity coding distribution defined in (16), the transition redundancy for any x_1^T with respect to transition pattern $\mathcal{T}$ satisfies the upper bound

$$\log \frac{P_c(x_1^T|\mathcal{T})}{P_c(x_1^T)} \leq C(\mathcal{T}) \log \frac{(T-1)e}{C(\mathcal{T})} + \frac{1}{2}\log(T-1) + 1 \tag{23}$$

for all transition patterns $\mathcal{T}$ over T symbols. The bound holds also for $C(\mathcal{T}) = 0$ if we follow the convention $z \log z = 0$ for $z = 0$.

Proof: From (18) we know that

$$P_{\rm tr}(\mathcal{T}) = P_e(T-1-C(\mathcal{T}), C(\mathcal{T})) \tag{24}$$

if $\mathcal{T}$ is a transition pattern when the source generates T symbols. In order to find a useful bound for $\log P_{\rm tr}(\mathcal{T})$ observe first that for nonnegative integers u and v with $u+v \geq 1$ by the bound in (5) we get

$$\begin{aligned} \log P_e(u,v) \geq\ &u \log \frac{u}{u+v} + v \log \frac{v}{u+v} \\ &- \frac{1}{2}\log(u+v) - 1. \end{aligned} \tag{25}$$

For the first term on the right-hand side of this inequality, we can write, using the basic inequality $\ln z \leq z - 1$, that

$$\begin{aligned} u \log \frac{u}{u+v} &= \frac{-u}{\ln 2}\cdot \ln \frac{u+v}{u} \geq \frac{-u}{\ln 2}\cdot\frac{v}{u} \\ &= \frac{-v}{\ln 2} = -v \log e. \end{aligned} \tag{26}$$

Assuming that $u \log u = 0$ for $u = 0$, we can see that this inequality also holds for $u = 0$. Combining all this leads to

$$\log P_e(u,v) \geq -v \log \frac{(u+v)e}{v} - \frac{1}{2}\log(u+v) - 1. \tag{27}$$

For any sequence $x_1^T \in \{0,1\}^T$ the *transition redundancy* can now be upper-bounded as in (10), i.e.,

$$\log \frac{P_c(x_1^T|\mathcal{T})}{P_c(x_1^T)} \leq \log \frac{1}{P_{\rm tr}(\mathcal{T})}, \quad \text{for all } \mathcal{T}. \tag{28}$$

Substitution of $u = T - 1 - C(\mathcal{T})$ and $v = C(\mathcal{T})$ in (27) yields the theorem. Note that $u + v = T - 1 \geq 1$. □

The theorem says that the transition redundancy is bounded by a bias term which is roughly $\frac{1}{2}\log T$ bits and then for each transition roughly $\log T$ additional bits. The bias term can be considered a consequence of not knowing in advance the number of transitions in the sequence. The additional terms seem to be necessary since the decoder needs (approximations of) the coordinates of the transitions.

Example (Continued): The transition redundancy for our example source generating the sequence x_1^T, with $T = 1000$ is, according to Theorem 1, upper-bounded by

$$3 \log \frac{999e}{3} + \frac{1}{2}\log 999 + 1 = 35.45 \quad \text{bits.}$$

For the generated sequence we have also *computed* the transition redundancy

$$\log(P_c(x_1^t|\mathcal{T})/P_c(x_1^t)), \quad \text{for } t = 1, T.$$

This redundancy is plotted in Fig. 3. It satisfies the bound 35.45 bits as we can see. Moreover, we see that for each transition we get a sharp increase of this redundancy. We can explain this by noting that the decoder must be informed almost immediately about these transitions and this requires roughly $\log t$ bits for a transition that occurred at time t. It may not be necessary, however, to transmit the exact time to the decoder, especially if the parameter change is small.

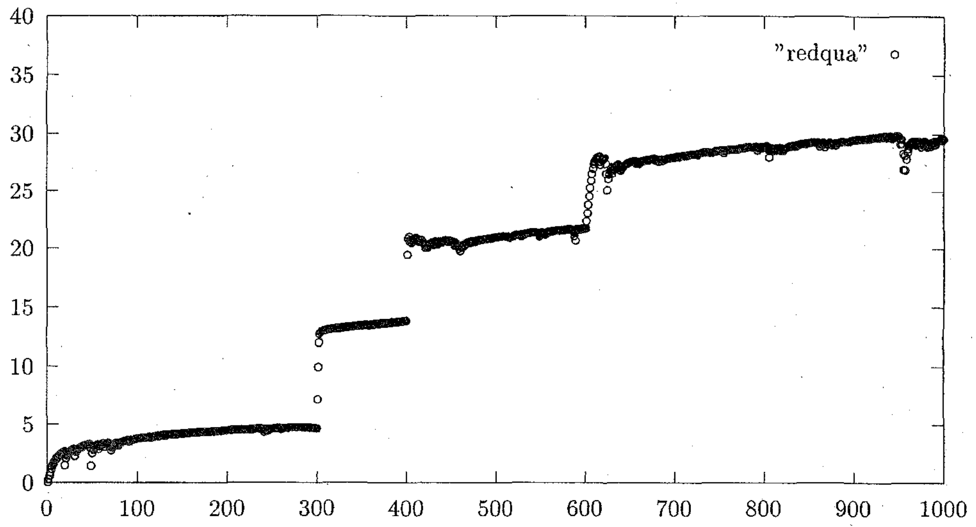


Fig. 3. Redundancy for the quadratical method as a function of t for $t = 1, 2, \cdots, 1000$.

Implementation: The quadratical-complexity coding distribution suggested in Definition 3 leads to the following implementation. Suppose that in all existing nodes (records) $(t-1, r, s)$ the probabilities $P(x_1^{t-1}, (t-1, r, s))$ and the counts $a(x_s^{t-1})$ and $b(x_s^{t-1})$ are stored. The probabilities $P(x_1^t, (t, p, q))$ can now be determined using (15) and stored in nodes (t, p, q). Summing over all these nodes as in (16) gives $P_c(x_1^t)$. The counts $a(x_q^t)$ and $b(x_q^t)$ in record (t, p, q) are determined using the counts $a(x_q^{t-1})$ and $b(x_q^{t-1})$ in record $(t-1, p, q)$. For example, if $x_t = 1$ count $a(x_q^t) := a(x_q^{t-1})$ and $b(x_q^t) := b(x_q^{t-1}) + 1$ for all $q < t$ and $a(x_q^t) := 0$ and $b(x_q^t) := 1$ for $q = t$. If $x_t = 0$, we increase the a-counts instead of the b-counts. After this the records $(t-1, r, s)$ can be deleted.

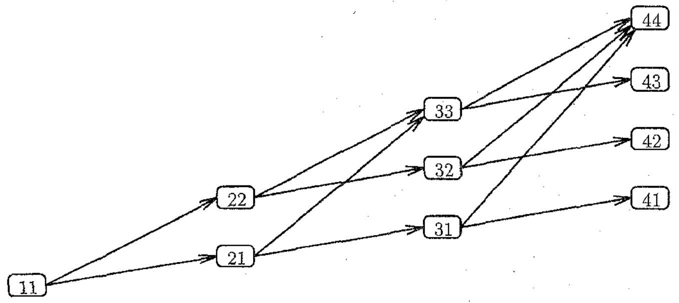
Storage Complexity: Since for processing symbol x_t there are $(t^2 - t + 2)/2$ nodes that need to be stored (while the nodes corresponding to x_{t-1} are deleted), the *instantaneous storage complexity* grows *quadratically* in T.

Computational Complexity: To determine the total number of computations (multiplications) to process x_1^T, note that there are two ways to leave the record $(t-1, r, s)$. Each of them involves a multiplication. If a transition is assumed, the next record is $(t, r+1, t)$, otherwise, the next record is (t, r, s). The number of computations related to processing x_t for $t > 1$ is, therefore, twice the number of records that are stored for processing x_{t-1}, i.e., $(t-1)^2 - (t-1) + 2 = t^2 - 3t + 4$. The total number of computations for processing x_1^T follows from summing the terms corresponding to x_t for $t = 2, T$, which appears to be $(T^3 - 3T^2 + 8T - 6)/3$. Therefore, there is a *cubic* dependency between the *total computational complexity* and the source sequence length T.

III. A Linear-Complexity Coding Method

The quadratical-complexity coding method gives a nice performance as we have seen in the previous section, but the complexity of this method is also quite large. Therefore, we will discuss a method here that has a storage complexity which is linear in the sequence length T. Although in theory the (transition) redundancy resulting from this method is higher than the redundancy for the quadratical method, the differences turn out to be small in practise. We will now assume that the source follows a path in a *linear* transition diagram (see Fig. 4).

Definition 4: The source now starts in state $(1, 1)$ of the linear transition diagram and generates the first output x_1 according to



$$\begin{tikzcd}[row sep=small] &&& \boxed{44} \\ && \boxed{33} \arrow[ur] \arrow[r] & \boxed{43} \\ & \boxed{22} \arrow[ur] \arrow[r] & \boxed{32} \arrow[uur] \arrow[r] & \boxed{42} \\ & \boxed{21} \arrow[uur] \arrow[r] & \boxed{31} \arrow[uuur] \arrow[r] & \boxed{41} \\ \boxed{11} \arrow[uur] \arrow[ur] &&& \end{tikzcd}$$

Fig. 4. Linear transition diagram for $t = 1, 2, 3, 4$. State (t, t_c) is now denoted as tt_c.

parameter θ_0. When the source produces output x_t, it is in a state (t, t_c), which indicates that the last time index *before* which a change occurred is t_c. The current parameter is θ_c. As in (13), we assign the following probabilities to the values 0 and 1 that can be generated:

$$P_c(X_t = 0|(t, t_c)) = \frac{a(x_{t_c}^{t-1}) + 1/2}{t - t_c + 1}$$
$$P_c(X_t = 1|(t, t_c)) = \frac{b(x_{t_c}^{t-1}) + 1/2}{t - t_c + 1}. \tag{29}$$

After having generated x_t, the parameter can change and the source will go to state $(t+1, t+1)$. If the parameter remains the same, the next state of the source will be $(t+1, t_c)$. We now assign the following probabilities to these two alternatives to obtain the Krichevsky–Trofimov estimators:

$$P_{\mathrm{tr}}((t+1, t_c)|(t, t_c)) = \frac{t - t_c + 1/2}{t - t_c + 1}$$
$$P_{\mathrm{tr}}((t+1, t+1)|(t, t_c)) = \frac{1/2}{t - t_c + 1} \tag{30}$$

where we should note that always $1 \leq t_c \leq t$. The source output x_t and the next state $(t+1, q)$ where $q = t_c$ or $t+1$ are independent of each other given the current state (t, t_c). Note that now the conditional coding probabilities in (29) and the conditional transition probabilities in (30) depend solely on t_c and not on c.

We will now specify the linear-complexity coding distribution over the source sequences.

Definition 5: For each sequence $x_1^T \in \{0,1\}^T$ we compute for all $x_1^t \in \{0,1\}^t, t = 1, T$, the probabilities $P_c(x_1^t, (t, t_c))$ recursively, i.e.,

$$P_c(x_1^t,(t,q)) \triangleq \sum_s P_c(x_1^{t-1},(t-1,s)) \cdot P_{\mathrm{tr}}((t,q)|(t-1,s))P_c(x_t|(t,q)) \quad (31)$$

for $t>1$, and $P_c(x_1,(1,1)) = 1/2$ for both $x_1 = 0$ and 1. The summation is only over s such that $1 \le s \le t-1$.

The coding probabilities

$$P_c(x_1^t), x_1^t \in \{0,1\}^t, \quad t = 1,T$$

are now defined as follows:

$$P_c(x_1^t) \triangleq \sum_q P_c(x_1^t,(t,q)). \quad (32)$$

Here we assume that the summation is only over q such that $1 \le q \le t$. Of course, $P_c(\phi) = 1$. □

Just like for the quadratical method, we can demonstrate that also the linear-complexity coding distribution is a weighting of all coding distributions $P_c(x_1^T|\mathcal{T})$ over all transition patterns $\mathcal{T}$ over T source symbols, i.e.,

$$P_c(x_1^T) = \sum_q \sum_{\mathcal{T}\to(T,q)} P_{\mathrm{tr}}(\mathcal{T})P_c(x_1^T|\mathcal{T}) \quad (33)$$

where

$$\sum_{\mathcal{T}\to(T,q)}$$

means a summation over all $\mathcal{T}$ that lead to state (T,q). The probability $P_c(x_1^T|\mathcal{T})$ is as defined in (8).

Furthermore, for a transition pattern $\mathcal{T}$ such that $\mathcal{T} \to (T,q)$ for some q, it follows from inspection that

$$P_{\mathrm{tr}}(\mathcal{T}) = \left(\prod_{c=0,C(\mathcal{T})-1} P_e(t_{c+1}-t_c-1,1)\right) P_e(T-t_{C(\mathcal{T})},0) \quad (34)$$

where the t_c for $c = 1, C(\mathcal{T})$ are the transition times in $\mathcal{T}$. This probability is composed out of $C(\mathcal{T})+1$ Krichevsky–Trofimov estimators, the first $C(\mathcal{T})$ of them "end" after a transition occurs, the last one "continues" until the end of the sequence.

The next theorem gives an upper bound on the transition redundancy for the linear method.

Theorem 2: Let $T \ge 2$. If we use the linear coding distribution defined in (32), the transition redundancy for any x_1^T with respect to pattern $\mathcal{T}$ satisfies the upper bound

$$\log \frac{P_c(x_1^T|\mathcal{T})}{P_c(x_1^T)} \le \frac{3C(\mathcal{T})}{2}\log\frac{T-1}{C(\mathcal{T})} + \frac{1}{2}\log(T-1) + C(\mathcal{T})\log(2e) + 1 \quad (35)$$

for all transition patterns $\mathcal{T}$ over T symbols. The bound holds also for $C(\mathcal{T}) = 0$ if we follow the convention $z \log z = 0$ for $z = 0$.

Proof: The probability $P_{\mathrm{tr}}(\mathcal{T})$ assigned to pattern $\mathcal{T}$ is given by (34). By the lower bound in (5) and by the basic inequality $\ln z \le z-1$ we obtain that

$$\begin{aligned}\log P_e(u-1,1) &\ge (u-1)\log\frac{u-1}{u} + \log\frac{1}{u} - \frac{1}{2}\log u - 1\\ &= -\frac{u-1}{\ln 2}\ln\frac{u}{u-1} - \frac{3}{2}\log u - 1\\ &\ge -\frac{u-1}{\ln 2}\frac{1}{u-1} - \frac{3}{2}\log u - 1\\ &= -\log e - \frac{3}{2}\log u - 1\\ &= -\frac{3}{2}\log u - \log(2e) \end{aligned} \quad (36)$$

for integers $u \ge 1$. For integers $v \ge 1$ the lower bound in (5) yields

$$\log P_e(v,0) \ge -\frac{1}{2}\log v - 1 \quad (37)$$

while for $v = 0$ we obtain $\log P_e(v,0) = 0$. Substituting these bounds in $\log(1/P_{\mathrm{tr}}(\mathcal{T}))$, where $P_{\mathrm{tr}}(\mathcal{T})$ is as in (34), yields two terms. The first term corresponds to the $C(\mathcal{T})$ transitions. Assuming that $C(\mathcal{T}) > 0$ and using (36) we get for this first term

$$\begin{aligned}&\sum_{c=0,C(\mathcal{T})-1} \frac{3}{2}\log(t_{c+1}-t_c) + C(\mathcal{T})\log(2e)\\ &= \frac{3C(\mathcal{T})}{2}\sum_{c=0,C(\mathcal{T})-1}\frac{1}{C(\mathcal{T})}\log(t_{c+1}-t_c) + C(\mathcal{T})\log(2e)\\ &\le \frac{3C(\mathcal{T})}{2}\log\left(\frac{1}{C(\mathcal{T})}\sum_{c=0,C(\mathcal{T})-1}(t_{c+1}-t_c)\right) + C(\mathcal{T})\log(2e)\\ &= \frac{3C(\mathcal{T})}{2}\log\frac{t_{C(\mathcal{T})}-1}{C(\mathcal{T})} + C(\mathcal{T})\log(2e)\\ &\le \frac{3C(\mathcal{T})}{2}\log\frac{T-1}{C(\mathcal{T})} + C(\mathcal{T})\log(2e)\end{aligned} \quad (38)$$

where we used the ∩-convexity of the log in the first inequality and the fact that $t_{C(\mathcal{T})} \le T$ in the second. It is important to note that $t_{c+1} > t_c$ for $c = 0, C(\mathcal{T})-1$. Note also that since $z\log z = 0$ for $z = 0$, the bound on the first term holds for $C(\mathcal{T}) = 0$.

For the second term in the bound for $\log(1/P_{\mathrm{tr}}(\mathcal{T}))$ we get using (37) for $T > t_{C(\mathcal{T})}$ that

$$\frac{1}{2}\log(T-t_{C(\mathcal{T})}) + 1 \le \frac{1}{2}\log(T-1) + 1 \quad (39)$$

since $t_{C(\mathcal{T})} \ge t_0 = 1$. If $T = t_{C(\mathcal{T})}$, the second term is zero, which is upper-bounded by $\frac{1}{2}\log(T-1)+1$.

Combining all this we can conclude that

$$\log\frac{1}{P_{\mathrm{tr}}(\mathcal{T})} \le \frac{3C(\mathcal{T})}{2}\log\frac{T-1}{C(\mathcal{T})} + \frac{1}{2}\log(T-1) + C(\mathcal{T})\log(2e) + 1 \quad (40)$$

which yields the theorem. □

The theorem says that the transition redundancy is now bounded by a bias term which is roughly $\frac{1}{2}\log T$ bits and then for each transition roughly $\frac{3}{2}\log T$ additional bits. We see that each transition costs roughly 50% more than for the quadratical method.

Example (Continued): The transition redundancy for the linear method for our example source is upper-bounded by

$$\frac{9}{2}\log\frac{999}{3} + \frac{1}{2}\log 999 + 3\log(2e) + 1 = 51.02 \quad \text{bits.}$$

This transition redundancy is plotted in Fig. 5. As we can observe the bound 51.02 bist is satisfied. Moreover, it appears that the measured transition redundancy is not as close to the upper bound as for the quadratical method.

Implementation: The coding distribution suggested in Definition 5 leads to an implementation in which in all existing nodes (records) $(t-1,s)$ the probabilities $P(x_1^{t-1},(t-1,s))$ and the counts $a(x_s^{t-1})$ and $b(x_s^{t-1})$ are stored. The probabilities $P(x_1^t,(t,q))$ are now determined using (31) and stored in nodes (t,q). Summing over all these nodes as in (32) gives $P_c(x_1^t)$. The counts $a(x_q^t)$ and $b(x_q^t)$ in record (t,q) are determined using the counts $a(x_q^{t-1})$ and $b(x_q^{t-1})$ in record $(t-1,q)$ just as in the quadratical method. After updating, the records $(t-1,s)$ can be deleted.

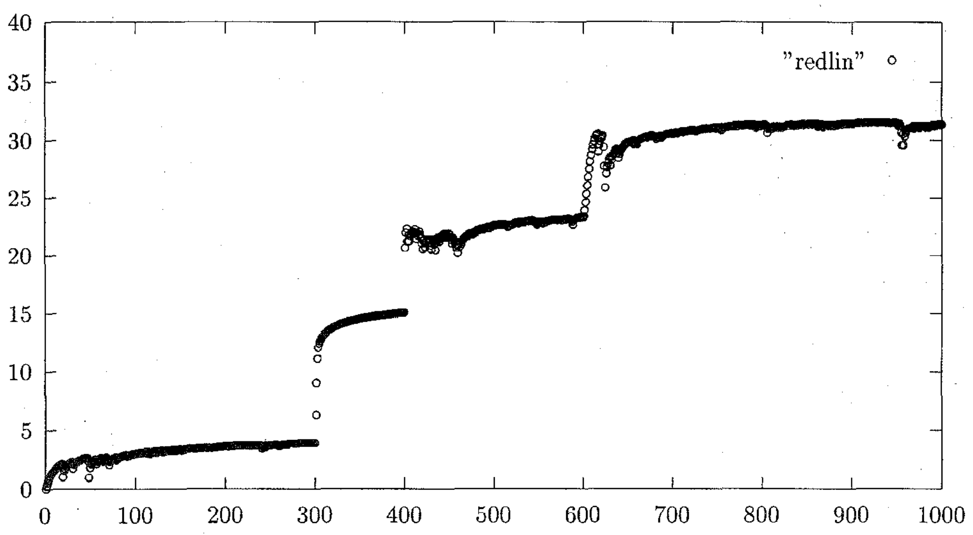


Fig. 5. Redundancy for the linear method as a function of t for $t = 1, 2, \cdots, 1000$.

Storage Complexity: Since for symbol x_t there are t nodes that need to be stored (while the nodes corresponding to x_{t-1} are deleted), the *instantaneous storage complexity* grows *linearly* in T.

Computational Complexity: To compute the total number of computations (multiplications) to process x_1^T, note that there are again two ways to leave the record $(t-1, s)$. Each of them involves a multiplication. If a transition is assumed, the next record is (t, t), otherwise, the next record is (t, s). The number of computations related to processing x_t for $t > 1$ is, therefore, twice the number of records that are stored for processing x_{t-1}, i.e., $2(t-1)$. The total number of computations for processing x_1^T follows from summing the terms corresponding to x_t for $t = 2, T$, which appears to be $T^2 - T$. Therefore, there is a *quadratic* dependency between the *total computational complexity* and the source sequence length T for the linear-complexity coding method.

IV. Conclusion

Merhav [3] investigated compression techniques for sources whose parameters can change abruptly at unknown points. He showed that for any uniquely decodable code the redundancy is lower-bounded by $\frac{1}{2}\log T$ bits for each parameter (parameter redundancy) plus $\log T$ bits for each transition (transition redundancy). Moreover, it is outlined by Merhav how to achieve this lower bound with a strongly sequential universal code, i.e., a code that does not need knowledge of T.

Actually, Merhav treated the achievability case where there is only one transition. The extension to more transitions is straightforward, he claims. However, if the number of transitions is not known in advance, the Merhav procedure can become quite complex since it involves weighting over the coding distributions $P_c(x_1^T|\mathcal{T})$ corresponding to all transition patterns $\mathcal{T}$ of sequence x_1^T. Note that there are 2^{T-1} such patterns, hence this would lead to a complexity which is exponential in T.

In contrast to Merhav, we concentrated on finding explicit coding methods for the case where the number of transitions is unrestricted and that do not have exponential complexity behavior. We suggested two coding methods, the quadratical-complexity method and the linear-complexity method, both based on the concept of weighting (see, e.g., Ryabko [7], [6] (but more recently also [10]) and Weinberger *et al.* [9]). Both methods are analyzed and it was shown that the quadratical method achieves the Merhav lower bound in the sense that each additional transition gives a redundancy increase of roughly $\log T$ bits. The linear method has a slightly larger transition redundancy ($\frac{3}{2}\log T$ bits per additional transition) but also a more acceptable complexity.

Both methods have a bias term contributing to the transition redundancy which is roughly $\frac{1}{2}\log T$ bits. This bias term is the result of not knowing the number of transitions in advance. From simulations we could see that in practice the redundancies of both methods are very close.

Just like Merhav, we would like to point out that this source coding situation should be treated as a Minimum Description Length (MDL) problem (see Rissanen [4], [5]). What the transmitter has to convey to the receiver is a specification (code) of the transition pattern (the model in terms of [10]) plus the code for the source sequence given the pattern. The last code contains a term coming from the fact that we do not know the parameters (parameter redundancy). The pattern is chosen in such a way that the cost of specifying the pattern plus the code length of the source sequence given the pattern, is minimized (see also (12)). We are not making a definite choice here, instead we are weighting over all transition patterns such that only the best pattern survives.

Acknowledgment

The author acknowledges discussions with Dr. Y. Shtarkov about the relation between the present problem and the switching source. The switching source was intensively studied by Shtarkov in [8].

The author wishes to thank F. Casadei, a student from the University of Bologna, Italy, who contributed to this research project. He also wishes to thank a reviewer and the Associate Editor M. Feder for valuable comments.

References

[1] F. Jelinek, *Probabilistic Information Theory* New York: McGraw-Hill, 1968, pp. 476–489.
[2] R. E. Krichevsky and V. K. Trofimov, "The performance of universal encoding," *IEEE Trans. Inform. Theory*, vol. IT-27, no. 2, pp. 199–207, Mar. 1981.
[3] N. Merhav, "On the minimum description length principle for sources with piecewise constant parameters," *IEEE Trans. Inform. Theory*, vol. 39, no. 6, pp. 1962–1967, Nov. 1993.
[4] J. Rissanen, "Universal coding, information, prediction, and estimation," *IEEE Trans. Inform. Theory*, vol. IT-30, no. 4, pp. 629–636, July 1984.
[5] ——, *Stochastic Complexity in Statistical Inquiry.* Singapore: World Scientific, 1989.

[6] B. Ya. Ryabko, "Twice-universal coding," *Probl. Inform. Transm.*, vol. 20, no. 3, pp. 24–28, July–Sept. 1984.
[7] ——, "Prediction of random sequences and universal coding," *Probl. Inform. Transm.*, vol. 24, no. 2, pp. 3–14, Apr.–June 1988.
[8] Yu. M. Shtarkov, "Switching discrete sources and its universal encoding," *Probl. Inform. Transm.*, vol. 28, no. 3, pp. 95–111, July–Sept. 1992.
[9] M. J. Weinberger, N. Merhav, and M. Feder, "Optimal sequential probability assignment for individual sequences," *IEEE Trans. Inform. Theory*, vol. 40, no. 2, pp. 384–396, Mar. 1994.
[10] F. M. J. Willems, Y. M. Shtarkov, and Tj. J. Tjalkens, "The context-tree weighting method: Basic properties," *IEEE Trans. Inform. Theory*, vol. 41, no. 3, pp. 653–664, May 1995.

Fault-Tolerant Cube Graphs and Coding Theory

Jehoshua Bruck, *Senior Member, IEEE*, and Ching-Tien Ho, *Member, IEEE*

Abstract—**Hypercubes, meshes, tori, and Omega networks are well-known interconnection networks for parallel computers. The structure of those graphs can be described in a more general framework called cube graphs. The idea is to assume that every node in a graph with q^ℓ nodes is represented by a unique string of ℓ symbols over GF (q). The edges are specified by a set of *offsets*, those are vectors of length ℓ over GF (q), where the two endpoints of an edge are an offset apart. We study techniques for tolerating edge faults in cube graphs that are based on adding redundant edges. The redundant graph has the property that the structure of the original graph can be maintained in the presence of edge faults. Our main contribution is a technique for adding the redundant edges that utilizes constructions of error-correcting codes and generalizes existing *ad hoc* techniques.**

Index Terms—**Fault tolerance, parallel computing, interconnection networks, hypercubes, omega networks, error-correcting codes.**

I. Introduction

The mesh, torus, hypercube, and Omega networks are some of the most important interconnection networks for parallel computers. One of the most important issues in the design of a system which contains many components is the system's performance in the presence of faults. Hence, it is of major practical importance to develop efficient techniques (in terms of the cost of the redundancy) to handle faults in those architectures.

Our approach for handling faults is based on a graph model. In this model the architecture is viewed as a graph, where the nodes represent the processors and the edges represent communication links between the nodes. A target graph is selected and the required amount of fault tolerance f is determined. Then a fault-tolerant graph, that has the same number of nodes as the target graph, is defined with the property that given *any* set of f or fewer faulty edges, the remaining graph (after removal of the faulty edges) is guaranteed to contain the target graph as a subgraph. Note that this approach guarantees that any algorithm designed for the target graph will run *with no slowdown* in the presence of f or fewer edge faults in the fault-tolerant graph, regardless of their distribution. Minimizing the cost in this model amounts to constructing a fault-tolerant graph with minimum degree. (See [4] for constructions that address node faults.)

Manuscript received September 3, 1995; revised February 16, 1996. This work was supported in part by the NSF Young Investigator Award CCR-9457811, by the Sloan Research Fellowship and under a Grant from the IBM Almaden Research Center, San Jose, CA.

J. Bruck is with the California Institute of Technology, MS 136-93, Pasadena, CA 91125 USA.
C.-T. Ho is with the IBM Almaden Research Center, San Jose, CA 95120 USA.

Publisher Item Identifier S 0018-9448(96)04824-9.

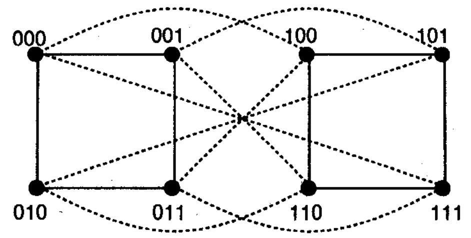


Fig. 1. A three-dimensional folded hypercube.

The key to our fault-tolerant constructions is a technique based on error-correcting codes for adding redundant edges using the so-called *wildcard dimensions*. Our technique is capable of handling an arbitrary number of faults over arbitrary alphabets. This is a generalization of the construction in [5] that is limited to the case of MDS (Maximum Distance Separable) codes. The key to our generalization is proving the equivalence between the constructions of fault-tolerant cube graphs and constructions of generator matrices of error-correcting codes (while in [5] the construction is based on parity check matrices of error-correcting codes.) First, we will describe the technique for constructing fault-tolerant hypercubes, then we will explain how it can be extended to Omega networks, tori, meshes, and cube-connected cycles (CCC).

Let Q_ℓ denote the ℓ-dimensional hypercube. It consists of 2^ℓ nodes, where node i, $0 \leq i \leq 2^\ell - 1$, is represented by the ℓ-bit binary representation of i. Two nodes, say $X = (x_{\ell-1}x_{\ell-2}\cdots x_0)$ and $Y = (y_{\ell-1}y_{\ell-2}\cdots y_0)$, are connected by an edge iff there is a single j for which $x_j \neq y_j$, and this edge is called a *dimension-j* edge. Hence, the set of edges in the hypercube can be partitioned into ℓ dimensions.

In [9], it was suggested to add another set of edges to the hypercube, called the *wildcard dimension*, resulting in a *folded* hypercube. The folded hypercube topology was also defined independently in [7], but with a different name (the bisectional interconnection network) and with a different addressing scheme for the nodes. A formal definition of the folded hypercube is given next.

Definition 1: An ℓ-dimensional *folded* hypercube, denoted by F_ℓ, is an ℓ-dimensional hypercube to which extra links are added connecting every pair of nodes that are bit-wise complements of each other.

A *wildcard edge* in F_ℓ is an edge that connects node $X = (x_{\ell-1}x_{\ell-2}\cdots x_0)$ and node $\overline{X} = (\overline{x}_{\ell-1}\overline{x}_{\ell-2}\cdots\overline{x}_0)$. For example, F_3 is depicted in Fig. 1; notice that the neighbors of node (000) are (001), (010),(100), and (111). The structure of Q_ℓ and F_ℓ can be described using a more general framework, see also [8].

Definition 2: Let ℓ be a positive integer and let q be a prime number. Let $S \subseteq \{0, 1, \cdots, q-1\}^\ell$. The graph $N(q, \ell, S)$ is a graph with q^ℓ nodes and with the edges specified by the set of vectors in S. Any two nodes, say X and Y, are connected by an edge iff there exists a vector $V \in S$ such that $X + V = Y$ where addition is vector addition performed over GF (q).