

# Making the correct mistakes

Dharmendra S. Modha<sup>1</sup> and Narayana P. Santhanam<sup>2</sup>

## Abstract

We propose a new sequential, adaptive, quadratic-time algorithm for variable-rate lossy compression of memoryless sources at a fixed distortion. The algorithm uses approximate pattern matching and is modeled after the Lempel-Ziv algorithm. As a key new idea, the algorithm uses *lower mutual information* to carefully select “good” codewords. For Bernoulli sources with Hamming distortion, we empirically demonstrate that the algorithm (a) discovers the optimal reproduction type, (b) leads to absence of multiple matches, and (c) seems to approach the rate-distortion coding rate. Based on empirical observations, we formulate two conjectures that could imply that the algorithm is asymptotically optimal for memoryless sources.

## 1 Introduction

The central problem of lossy source coding is to find an universal (for stationary, ergodic sources), sequential, adaptive, and polynomial-time algorithm. The quest for such algorithms is important in theory as well in practice owing to broadband applications such as streaming multimedia, images, audio, cellular voice, and text. When no distortion is desired, the lossy coding problem simplifies to the well researched problem of lossless data compression where various well known algorithms such as dynamic Huffman coding, adaptive arithmetic coding, Lempel-Ziv algorithms, locally adaptive schemes, and grammar codes are known.

Unfortunately, no universal yet practically attractive algorithms are known for lossy compression. In fact, it is widely believed that no such algorithm exists. For example, [1, p. 2709] noted that “All universal lossy coding schemes found to date lack the relative simplicity that imbues Lempel-Ziv coders and arithmetic coders with economic viability. . . . This suggests it is unlikely that the “holy grail” of implementable universal lossy source coding will be discovered soon.”. When speaking about Lempel-Ziv-type algorithms for lossy data compression, [2, p. 2054] noted that “It seems, however, that low computational complexity is achievable only at the expense of yielding a nonoptimal distortion.” Also, [3] noted that “...it is our belief that a universal lossy source coding scheme with attractive computational complexity aspects will never be found.”

We propose a principled algorithm for lossy compression at a fixed distortion that is practical (sequential, adaptive, and quadratic-time), and exhibits tantalizing empirical evidence that suggests that the algorithm may be asymptotically optimal for memoryless sources. Our algorithm is in the spirit of the incremental parsing Lempel-Ziv (LZ78) algorithm [4] for lossless coding, and, when no distortion is desired, the algorithm simplifies to LZ78. Like the LZ78 algorithm, the algorithm *sequentially* parses the source sequence into non-overlapping phrases, mapping each phrase to a *codeword* in a *dictionary*, which in turn is updated sequentially. The per-letter distortion between any phrase and its associated codeword does not exceed the desired distortion. Typically, there are multiple ways to parse the incoming source string and map it into codewords. Multiple matches are a sign of underlying redundancy, and, hence, are a symptom of the fact that we are not operating near the rate-distortion curve. We focus on the following key question:

How to select between multiple parsings?

<sup>1</sup>IBM Research, 650 Harry Road, San Jose, CA 95120, 408-927-1887, dmodha@us.ibm.com

<sup>2</sup>Dept. of ECE, UCSD, 9500 Gilman Dr, La Jolla, CA 92093, 858-752-7871, nsanthan@ucsd.edu

The choice of a codeword affects the future parsing, since the chosen codeword is used to update the dictionary. Moreover, the codeword chosen affects the per-letter code length for the phrase in question. For example, if we encode a phrase using a uniform code over all the codewords in the dictionary, the longer the length of the phrase, the lower the per-symbol code length of the phrase. We would like to carefully choose the codeword that best balances between the per-letter code rate in the current epoch and the quality of the resulting codebook for future epochs. As our key new contribution, we accomplish this task by utilizing lower mutual information.

## 2 Prior Work

We present a representative, but necessarily brief and non-exhaustive review of various known lossy coding schemes, focussing on algorithmic results. For references to earlier results on existence of universal lossy codes involving exponential-time constructions, see, Kieffer [5].

Cheung and Wei [6] extended the move-to-front algorithm to lossy source coding. The algorithm is sup-optimal [7]. Later, Zhang and Wei [8] proposed an universal, on-line lossy coding algorithm for the fixed-rate case.

Another group of papers have focussed on lossy extensions of the Lempel-Ziv algorithm. The central idea is to use approximate string matching [9, 10] instead of exact string matching used in the Lempel-Ziv algorithms. Morita and Kobayashi [11] extended the LZW algorithm. The algorithm is known to be sub-optimal for memoryless sources [7]. Constantinescu and Storer [12, 13] combined ideas from lossless Lempel-Ziv algorithms and vector quantization to design first practical implementations of lossy image compression based on approximate pattern matching. The problem of “selecting amongst multiple matches” mentioned above was termed the “Match Heuristic” in their work; see, also, Storer [14, p. 111]. Later, Steinberg and Gutman [15] and Luczak and Szpankowski [16] considered the fixed-database version of the Lempel-Ziv algorithm, and provided sub-optimal performance guarantees. Finally, Yang and Kieffer [7] established that all previous fixed-database extensions of the Lempel-Ziv algorithm are suboptimal. Kontoyiannis [17] presented a scheme where multiple databases are used at the encoder and must also be known to the decoder. When the reproduction alphabet is large, the number of training databases is unreasonably large. Atallah et al. [18] considered a cubic-time, adaptive algorithm (PMIC) in the spirit of LZ77. Their algorithm is not sequential in the sense of [19], since its encoding delay grows faster than  $o(n)$ . Alzina et al. [20] combined ideas from [18] and [12, 13] to propose a 2D-PMIC algorithm that is more suited for 2D images. Continuing the quest for Lempel-Ziv-type lossy algorithms, Zamir and Rose [21] further studied the algorithm in [11]. From the multiple codewords that may match a source word, they suggest choosing one “at random”. From a theoretical perspective, by assuming uniqueness, Zamir and Rose [22] proposed a natural type selection scheme for finding the type of the optimal reproduction distribution. In later work, Kochman and Zamir [23] pointed out that the theoretical procedure in [22] is in itself not practical and demonstrated an application of natural-type selection to on-line codebook selection from a parametric class.

Along a different line, Yang and Kieffer [3] have proposed exponential-time Lempel-Ziv-type block codes that are universal (for stationary, ergodic sources and for individual sequences). In a related work, Yang and Zhang [24] presented fixed-slope universal lossy coding schemes that search for the reproduction sequence through a trellis in a fashion reminiscent of the Viterbi algorithm.

The current paper builds on the direction and the approach in [25, 26], but presents a completely novel codeword selection mechanism.

### 3 Preliminaries

#### Strings and Distortion

All logarithms are base 2. Let  $\mathbb{N}$  denote the set of natural numbers. Let  $\mathcal{A}$  denote a finite alphabet. We refer to elements of  $\mathcal{A}$  as *symbols* or *letters*, and  $|\mathcal{A}|$  is the number of letters. For  $m \in \mathbb{N}$ , let  $\mathcal{A}^m = \{(a_1, a_2, \dots, a_m) : a_i \in \mathcal{A}, 1 \leq i \leq m\}$  denote the set of all sequences (strings) of length  $m$  over  $\mathcal{A}$ . By convention,  $\mathcal{A}^0 = \{\lambda\}$ . Let  $\mathcal{A}^* = \cup_{m \geq 0} \mathcal{A}^m$  denote the set of all sequences of finite length over  $\mathcal{A}$ . For any string  $\bar{x} \in \mathcal{A}^*$ ,  $|\bar{x}|$  denotes its length, and  $\tau(\bar{x})$  its type. For example, if  $\mathcal{A} = \{0, 1\}$ , for the string  $\bar{x} = 001 \in \mathcal{A}^3$ ,

$$|\bar{x}| = 3 \text{ and } \tau(\bar{x}) = (2/3, 1/3).$$

By convention, every type also corresponds to a distribution. In the example above, the type  $(2/3, 1/3)$  is a distribution on  $\mathcal{A}$ , with probability of 0 being  $2/3$  and that of 1 being  $1/3$ . For  $i \leq j$ , we let  $a_i^j \equiv a_i, a_{i+1}, \dots, a_j$ . If  $j < i$ , then  $a_i^j$  is the empty string. A *dictionary* is a finite subset  $D \subset \mathcal{A}^*$ . A dictionary is termed *proper* if no string in the dictionary is a prefix of another. and it is *complete* if every one-sided infinite sequence has a prefix in the dictionary.

Let  $B$  and  $\tilde{B}$  denote finite source and reproduction alphabets, respectively. The *distortion measure* is a bounded, non-negative function  $d : B \times \tilde{B} \rightarrow \mathbb{R}$  such that for every  $b \in B$  there exists a  $\tilde{b} \in \tilde{B}$  such that  $d(b, \tilde{b}) = 0$ . For example, in case of binary alphabets,  $B = \tilde{B} = \{0, 1\}$  and  $d$  may simply be the Hamming distance. For  $m \in \mathbb{N}$ , we say that a sequence  $\tilde{b}_1^m \in \tilde{B}^m$  is a  $\epsilon$ -*match*,  $\epsilon \geq 0$ , of a sequence  $b_1^m \in B^m$  with respect to the single-letter distortion measure  $d_m$ , if  $\sum_{i=1}^m d(b_i, \tilde{b}_i) \leq m\epsilon$ .

#### Rate-Distortion

Let  $P$  denote a probability distribution on  $B$ . Let  $\mathcal{W}$  denote the set of all conditional distributions of the form  $W(y|x)$ , where  $y \in \tilde{B}$  and  $x \in B$ . Let  $D$  denote a desired distortion. The *rate-distortion* function is defined as

$$R(P, D) = \min_{W \in \mathcal{W}: d(P, W) \leq D} I(P, W),$$

where  $I(P, W)$  denotes the *mutual information* and  $d(P, W)$  denotes the average distortion induced by  $W$ . The fundamental significance of rate-distortion function stems from the fact that it is an asymptotic lower bound on the compression rate of codes operating at the fixed distortion  $D$ . Let  $W^*$  denote a *optimal forward channel* that achieves the rate-distortion function. The joint distribution on  $B \times \tilde{B}$  is then  $P(x)W^*(y|x)$ , where  $x \in B$  and  $y \in \tilde{B}$ . The *optimal reproduction distribution* is the induced distribution on  $\tilde{B}$ :

$$Q^*(y) = \sum_{x \in B} P(x)W^*(y|x)$$

#### Lower Mutual Information

**Lemma 1.** Let  $X$  denote a length- $n$  string picked from a memoryless source  $P^n$ . Then, for any string  $\bar{y}$  with type  $Q$ ,

$$-\frac{1}{n} \log P(\bar{y} \text{ is a } D\text{-match for } X) = I_m(P, Q, D) + o(1),$$

where the *lower mutual information* [8, 22] is

$$I_m(P, Q, D) \stackrel{\text{def}}{=} \inf_{W(y|x): \sum_{x \in B} P(x)W(y|x)=Q(y)} I(P, W). \quad \square$$

As one would expect from the lemma above, for a source  $P$ , the type of a string  $\bar{y}$  whose  $D$ -balls are most probable is  $Q^*$ , the optimal reproduction distribution of  $P$ . Intuitively speaking, codewords with the same type as the optimal reproduction distribution have the largest  $D$ -balls, hence, yield the best covering, and best coding. For a precise formulations of the above concepts, see [8, 22].  $I_m(P, Q, D)$  is a convex function of  $Q$  for a fixed  $P$ , with a minimum at  $Q^*$ . For a variety of distortion measures, it can be computed in closed form using the Kuhn-Tucker conditions for optimality since the mutual information is a convex function of the channel transition probabilities.

**Example 1.** Consider a binary, memoryless source with marginal  $P(1) \equiv p \leq 1/2$ ,  $\bar{p} = 1 - p$ , and let  $d$  be the Hamming distortion measure. Let us write,  $Q(1) \equiv q$ . If  $p > D$ , we obtain for

$$I_m(P, Q, D) = \begin{cases} h(q) - ph\left(\frac{q-d+p}{2p}\right) - \bar{p}h\left(\frac{q+d-p}{2\bar{p}}\right) & p - d \leq q \leq p + d, \\ \infty & \text{else,} \end{cases}$$

where  $h(\cdot)$  denotes the binary entropy function. Figure 3 plots  $I_m(P, Q, D)$  for various values of  $q$ , with  $p = 0.25$  and  $D = .1$ . The minimum is at  $(p - D)/(1 - 2D) = 0.1875$ .  $\square$

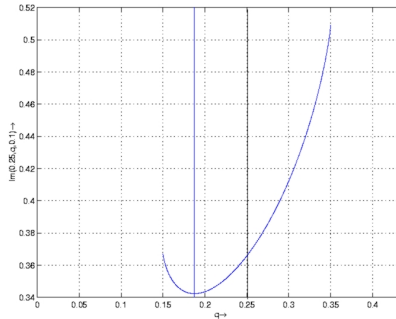


Figure 1: An example of lower mutual information.

## 4 Codelet parsing

We are now ready to describe our lossy compression algorithm. At the block level, the algorithm maps a source sequence  $x_1^n$  to a distorted sequence  $y_1^n$ , and then encodes and transmits the latter without loss using a LZ78 encoder. The Codelet parsing algorithm is described in Figure 2.

We think of the algorithm as going through epochs or iterations. In epoch  $i$ , a source phrase (a *sourcelet*) is consumed and a corresponding  $D$ -matching distorted phrase of the same length (a *codelet*) is emitted. The algorithm ensures that each new codelet is a one-letter extension of a previously emitted codelet, and, hence, the distorted sequence is naturally parsed using LZ78.

---

Input: source sequence  $x_1^n$ ,  $n \geq 1$ , target distortion  $D$ .

Initialize dictionary  $S_0 = \{\lambda\}$  and  $x_1^0 = y_1^0 = \lambda$ . Initialize iteration index  $i = 1$ .  
 Set  $\hat{t}(0) = \hat{t}(-1) = 0$ . /\*  $\hat{t}(i)$  denotes # symbols processed after  $i$  iterations.\*/  
 while ( $\hat{t}(i-1) < n$ )

1. Update dictionary by deleting the chosen codelet in the last step, and by adding all its one-letter extensions:

$$S_i = \left\{ S_{i-1} \setminus y_{\hat{t}(i-2)+1}^{\hat{t}(i-1)} \right\} \cup \left\{ y_{\hat{t}(i-2)+1}^{\hat{t}(i-1)} \circ \tilde{b} : \tilde{b} \in \tilde{B} \right\}.$$

2. Compute the set of codelets  $C_i \subseteq S_i$  that  $D$ -match the corresponding equi-length prefix of  $x_{\hat{t}(i-1)+1}^n$ . Since  $S_i$  is a complete and proper dictionary for every  $i$ ,  $C_i$  contains at least one element corresponding to a string in  $S_i$  that matches an equi-length prefix of  $x_{\hat{t}(i-1)+1}^n$  without any distortion.
3. From this set of matching codelets, choose

$$y_{\hat{t}(i-1)+1}^{\hat{t}(i)} = \arg \min_{v \in C_i} \left[ I_m \left( \tau \left( x_1^{\hat{t}(i-1)+|v|} \right), \tau(v), d \right) \right], \quad (1)$$

where  $\hat{t}(i)$  is also implicitly set above by the choice of the codelet in the dictionary and  $\tau(v)$  of a string  $v$  is the type of  $v$ . To break ties, pick the codelet with longest length, and among them, the codelet that occurred most recently.

4.  $i = i + 1$ .

endwhile

Write  $\hat{c}(x_1^n) = \hat{t}(i-2)$ . /\* # phrases parsed \*/

Output: The LZ78 parsing  $y_1^n \equiv y_1^{\hat{t}(1)}, y_{\hat{t}(1)+1}^{\hat{t}(2)}, \dots, y_{\hat{t}(\hat{c}(x_1^n)-1)+1}^{\hat{t}(\hat{c}(x_1^n))}, y_{\hat{t}(\hat{c}(x_1^n))+1}^n$ .

---

Figure 2: The Codelet Parsing Algorithm.

Step 3 contains the heart of the algorithm, and is a core new contribution of this paper. Typically,  $C_i$  will contain more than one  $D$ -match. Such excess matches are a sign of underlying redundancy. The main question is how to resolve this ambiguity, and select a good match. The choice of the codelet matters a great deal as we shall empirically illustrate in the next section. Intuitively, the longer the chosen codelet, the shorter the per-symbol codelength in the current epoch. However, the chosen codelet will change the codebook for future epochs and will affect future coding rate. Thus, a good code should strive to balance between the code rate in the current step versus the quality of the resulting codebook for the future. We select the codelet that has the lowest lower mutual information,  $I_m$ , that seemingly achieves this delicate balance as we now explain. Recall from Lemma 1 that the smaller the value of  $I_m$ , the higher the probability of the  $D$ -ball around it.

**future** A codelet with smaller  $I_m$  (and correspondingly larger  $D$ -ball) is more likely to get matched and extended in the future. Such a codelet will lead to longer codelets more quickly, and, hence, to a better coding rate in the future.

**present** Now, assign to each codelet a “probability” equal to the sum of probabilities of all semi-infinite sequences that would be mapped into it. In the absence of too many matching codelets (see Section 5), the entire  $D$ -ball around a codelet would map to it, and, hence, the sum would be inversely proportional to  $I_m$ . Thus, lower  $I_m$  will

lead to a better coding rate in the present epoch under an entropy coding scheme to transmit the codelets that uses these “probabilities”.

It is for this reason that we pick  $I_m$  to measure the goodness of a codelet.

## 5 Observations, Conjectures, and Analysis

At this point of research, we do not have a theoretical analysis of the algorithm. However, we make several observations that lead us to believe that codelet parsing is asymptotically optimal. In particular, we make two conjectures based on the observations, and explain how they would fit into a potential proof of optimality. We also empirically illustrate why a greedy algorithm that randomly picks one of the longest matches is unlikely to be asymptotically optimal, and contrast it to codelet parsing.

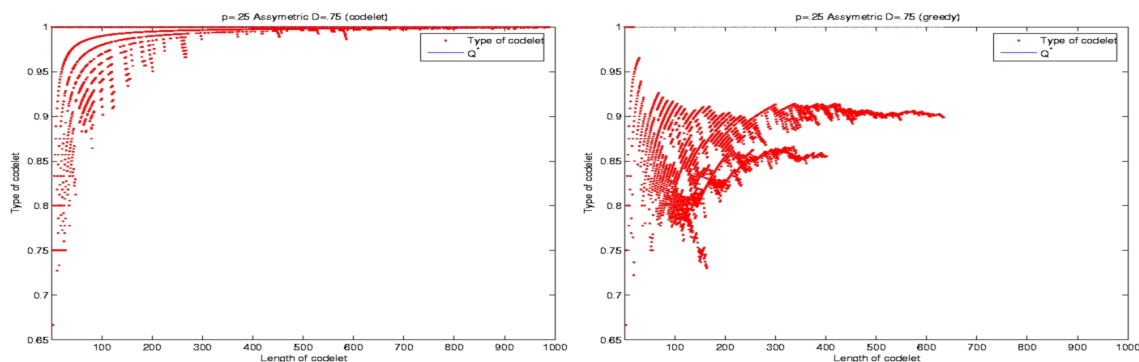


Figure 3: Codelet parsing (left panel) gravitates towards optimal reproduction type  $Q^*(1) = 1$ , while greedy (right panel) is seemingly lost. As a result, codelet parsing finds significantly longer phrases.

### Observation 1: Greedy is not enough

For this illustration, we used a binary, memoryless source with  $P(1) = .25$  and used an asymmetric distortion measure:  $d(0,0) = d(1,1) = 0, d(0,1) = 1, d(1,0) = 5$ . We chose  $D = 0.75$ . Clearly, the optimal reproduction type satisfies  $Q^*(1) = 1$ . In Figure 3, we plot the length of the codewords selected against their types (“fraction of ones”) for both codelet parsing and greedy (longest length codelet chosen at each step) algorithms. The algorithms are both run on the same 1M block of data. It can be clearly seen that codelet parsing gravitates towards the optimal reproduction type, whereas greedy algorithm fails to get close. This underscores our central thesis that careful selection of codewords is extremely important and that the simple greedy strategy is unlikely to be asymptotically optimal. The coding rates of codelet parsing and the greedy algorithm are 0.0365 and .04947 respectively.

### Observation 2: Type of codelets selected

We now exhibit<sup>3</sup> results using a random string of blocklength  $n = 10485760$  (10M) generated by a binary, memoryless source with  $P(1) = 0.1$ . From now on, we use the Hamming distortion. The reader may want to recall Example 3.1. We use distortion rates of 0.09 and 0.07 in the simulations, and the target rates are 0.0325 and 0.1031 respectively.

<sup>3</sup>The reported results are representative of the several experiments for various binary, memoryless sources that were carried out. The random data were collected from <http://www.random.org/>

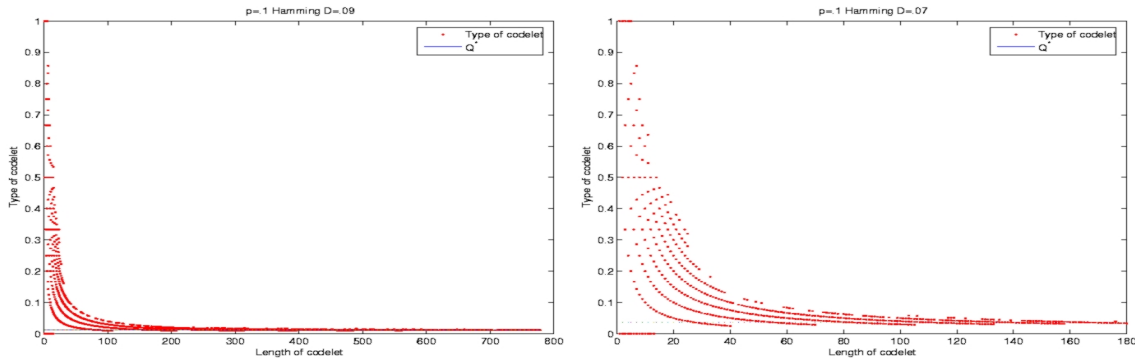


Figure 4: Longer codewords concentrate around the optimal reproduction type

In Figure 4, we plot the length of the selected codewords versus their types (“fraction of ones”) for two different distortion rates, namely,  $D = 0.09$  (left panel) and  $D = 0.07$  (right panel). In both cases, it can be seen that the longer codewords tend to concentrate near the optimal reproduction type. This suggests that codelet parsing adaptively discovers the optimal reproduction type empirically and keeps choosing codewords with *good types*, namely types for which  $I_m(p, q, d) \leq R(D) + .0175$  for the  $D = .09$  case, and  $I_m(p, q, d) \leq R(D) + .0479$  for the  $D = .07$  case. While Figure 4 shows that longer codewords seem to

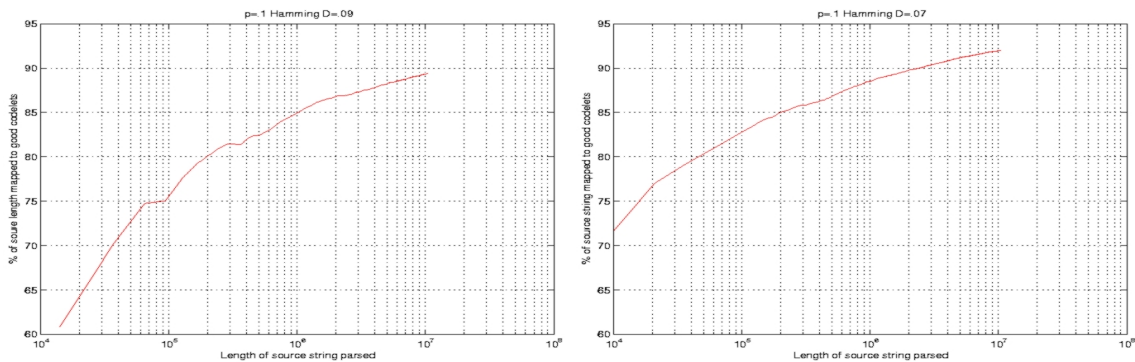


Figure 5: Most source words match codewords with good types.

have the correct type. Figure 5 shows a stronger result that most sourcelets match a codelet with good type, in other words, good codelets dominate and cover a large fraction of source words. Specifically, we notice that the percentage of the length of sourcelets that is parsed into codelets whose type  $Q$  satisfies  $I_m(P, Q, D) < R(D) + \epsilon$  seems to increase, getting close to 1. In the experiments above, we picked  $\epsilon = .0175$  for the  $D = 0.09$  case, and  $.0479$  for the  $D = .07$  case. These observations lead us to make the following conjecture.

**Conjecture 1.** For all  $\epsilon > 0$  and  $\delta > 0$ ,  $\exists L$  (large enough) such that if a codelet  $y$ ,  $|y| > L$ ,  $D$ -matches such an incoming *iid* sourcelet then:

$$\Pr \{I_m(P, \tau(y), D) < R(D) + \epsilon\} \geq 1 - \delta. \quad \square$$

### Observation 3: Number of competing matches

A large number of competing matches is a sign of sub-optimality, and lack thereof hints at optimality. In Figure 6 we plot the length of selected codelets versus the number of

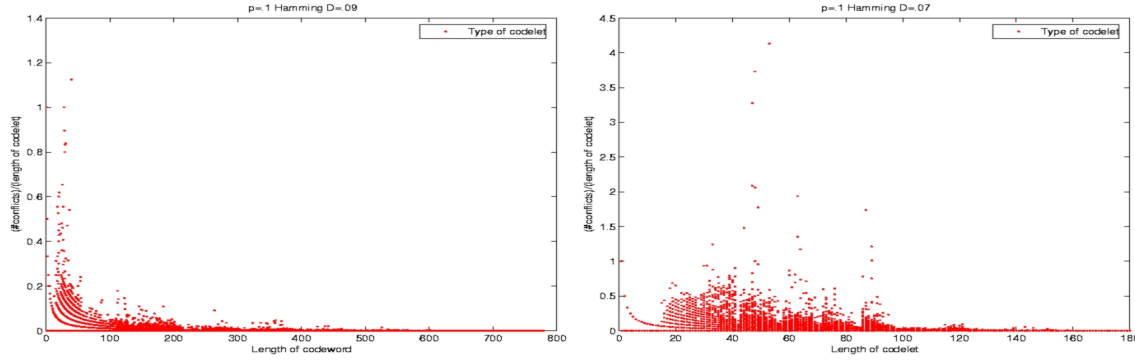


Figure 6: Longer codewords have good type and seem to be virtually unique.

competing codelets of the same type divided by the length of the selected codelet. This quantity approaches zero for  $D = 0.09$  (left panel) and  $D = 0.07$  (right panel), and, hence, the number of competing matches seem to be very small. Specifically, the phrase with the best type seems to be virtually unique.

We say that a source string  $x_1^n \in B^n$  that is parsed into  $\hat{c} \equiv \hat{c}(x_1^n)$  phrases has a  $\epsilon$ -good covering,  $\epsilon > 0$ , if in every epoch  $1 \leq i \leq \hat{c} - 1$  and for all  $y \in S_i$  such that  $I_m(P, \tau(y), D) < R(D) + \epsilon$ , we have that

$$\Pr \{ \text{a sourcelet maps to } y \} \geq \left[ 2^{-|y|(R(D)+\epsilon)} \right] / \text{poly}(|y|).$$

Let  $G_{n,\epsilon} \subset B^n$  denote the set of source strings that have  $\epsilon$ -good covering.

**Conjecture 2.** Fix  $\nu > 0$ . Then,  $\exists N$  such that for all  $n > N$

$$\Pr \{ x_1^n \in G_{n,\epsilon} \} \geq 1 - \nu. \quad \square$$

The actual coding rates obtained by simulation are 0.11841 and 0.270411 at distortions 0.085 and .061 when the target distortions are 0.09 and .07 respectively. As the blocklength increases, the actual distortion seems to increase towards the target  $D$  and the rate seems to decrease towards  $R(D)$ . We expect a slow convergence of the coding rate,  $\mathcal{O}\left(\frac{\log \log n}{\log n}\right)$ , similar to LZ algorithms. For perspective, observe that for  $n = 10M$ ,  $\frac{\log \log n}{\log n} = 0.1948$ .

## Towards Optimality

We now assume the above two conjectures are true, and outline an approach for a proof of optimality.

Fix  $\delta$ ,  $\epsilon$  and  $\nu$  (from Conjectures 1 and 2). A good codelet  $y$  has type satisfying  $I_m(P, \tau(y), D) < R(D) + \epsilon$ . Let  $G_{n,\epsilon}^c = B^n - G_{n,\epsilon}$ . The contribution to the expected code rate of all strings in  $G_{n,\epsilon}^c$  can be easily seen to be at most  $\nu$ , since the per-symbol code rate for each such string is at most 1, and the probability of  $G_{n,\epsilon}^c$  is  $< \nu$ .

We show that the contribution to the expected per-symbol code rate of all strings in  $G_{n,\epsilon}$  is approximately  $R(D)$ . To do so, we observe that for any string in  $B^n$ , all codelets with length  $\leq \log n - 2 \log \log n$  together contribute negligibly to the code rate. For a string in  $G_{n,\epsilon}$ , we observe that for all long codelets, the expectation of the instantaneous rate of the codelet, the number of bits used to describe it normalized by its length, is  $\approx R(D)$  bits.

**Codelets with length  $\leq \log n - 2 \log \log n$ :** Note that there can be at most  $2^{\log n - 2 \log \log n} = \frac{n}{\log^2 n}$  codelets with length  $< \log n - 2 \log \log n$ . Describing each one takes at most  $1 + \log n$

bits since there cannot be more than  $2n$  codelets from a length  $n$  string. Therefore, these codelets together contribute  $\frac{n}{\log^2 n} \cdot \log n$  bits to the codelength, which normalized by  $n$  yields the contribution to the rate,

$$\frac{n}{\log^2 n} \frac{\log n}{n} = \frac{1}{\log n}.$$

**Codelets with length  $> \log n - 2 \log \log n$ :** Let  $n$  be large enough that  $L$  in Conjecture 1 is less than  $\log n - 2 \log \log n$ . From Conjecture 1, the probability an the incoming phrase maps into a good codelet type is greater than  $1 - \delta$ .

For simplicity, we describe the codelet by first specifying its length  $l$ , its type  $Q$  among all possible  $l + 1$  types, followed by the lexicographic order of the codelet among all the length- $l$  codelets with type  $Q$  in the dictionary. If the incoming phrase maps to a good codelet, from Conjecture 2, there are at most  $\text{poly}(l)2^{l(R(D)+\epsilon)} \leq 2^{l(R(D)+2\epsilon)}$  codelets in the dictionary of that type for  $l$  (and hence  $n$ ) large enough. Therefore, this method of description requires

$$\log l + 2 \log \log l + 1 + \log(l + 1) + l(R(D) + 2\epsilon) \leq 2 \log l + 2 \log \log l + 2 + l(R(D) + 2\epsilon)$$

bits. If the incoming phrase does not map to a good codelet, we require at most  $\log n + 1$  bits to describe the previous phrase, since the number of codelets in the dictionary cannot exceed  $2n$ .

Since  $l = \Omega(\log n)$ , it follows that the expected instantaneous code rate is

$$(1 - \delta) \frac{2 \log l + 2 \log \log l + 2 + l(R(D) + \epsilon)}{l} + \delta \frac{\log n + 1}{l} = R(D) + 2\epsilon + \delta + \mathcal{O}\left(\frac{\log \log n}{\log n}\right).$$

It can be shown using elementary arguments that the contribution to the expected code rate of strings in  $G_{n,\epsilon}$  is also upper bounded by the above bound.

Combining the above with the contribution of strings in  $G_{n,\epsilon}^c$ , the expected coderate is upper bounded by

$$R(D) + 2\epsilon + \delta + \nu + \mathcal{O}\left(\frac{\log \log n}{\log n}\right).$$

Since  $\delta$ ,  $\epsilon$  and  $\nu$  are arbitrary, it follows that we can approach as close to  $R(D)$  as required.

The encoding of codelets used in the argument above is not the same as the one proposed. However, note that the number of bits to describe any good codelet in the above scheme is approximately the same, and we further believe that a uniform code (as proposed, and used in simulations) should be just as good.

## 6 Conclusions

We have presented a novel algorithm for variable-rate lossy compression of memoryless sources at a fixed distortion. The algorithm selects among multiple sequences within the allowed distortion to find one that is highly compressible. Based on empirical results, we conjecture that our algorithm or a suitable variant is a lossy analog of the Lempel-Ziv encoding.

## References

- [1] T. Berger and J. D. Gibson. Lossy source coding. *IEEE Trans. Inform. Theory*, 44(6):2693–2723, 1998.
- [2] A. D. Wyner, J. Ziv, and A. J. Wyner. On the role of pattern matching in information theory. *IEEE Trans. Inform. Theory*, 44(6):2045–2056, 1998.

- [3] En-Hui Yang and J. C. Kieffer. Simple universal lossy data compression schemes derived from the Lempel-Ziv algorithm. *IEEE Trans. Inform. Theory*, 42(1):239–245, 1996.
- [4] J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Trans. Inform. Theory*, 24(5):530–536, 1978.
- [5] J. C. Kieffer. A survey of the theory of source coding with a fidelity criterion. *IEEE Trans. Inform. Theory*, 39(5):1473–1490, 1993.
- [6] K. Cheung and V. K. Wei. A locally adaptive source coding scheme. In *Bilkent Conf. on New Trends in Comm., Cont., and Signal Proc.*, pages 1473–1482, 1990.
- [7] En-Hui Yang and J. C. Kieffer. On the performance of data compression algorithms based upon string matching. *IEEE Trans. Inform. Theory*, 44:47–65, 1998.
- [8] Z. Zhang and V. K. Wei. An on-line universal lossy data compression algorithm via continuous codebook refinement—part i: Basic results. *IEEE Trans. Inform. Theory*, 42(3):803–821, 1996.
- [9] M. J. Atallah, F. Chyzak, and P. Dumas. A randomized algorithm for approximate string matching. *Algorithmica*, 29:468–486, 2001.
- [10] G. Navarro. A guide to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.
- [11] H. Morita and K. Kobayashi. An extension of LZW coding algorithm to source coding subject to a fidelity criterion. In *Proc. 4th Joint Swedish-Soviet Int. Workshop on Information Theory, Gotland, Sweden*, pages 105–109, 1989.
- [12] C. Constantinescu and J. A. Storer. On-line adaptive vector quantization with variable size codebook entries. In *Data Compression Conf.*, pages 32–41, 1993.
- [13] C. Constantinescu and J. A. Storer. Improved techniques for single-pass vector quantization. *Proceedings of the IEEE*, 82(6):933–939, 1994.
- [14] J. A. Storer. *Data Compression: Methods and Theory*. Computer Science Press, Rockville, Maryland, 1988.
- [15] Y. Steinberg and M. Gutman. An algorithm for source coding subject to a fidelity criterion, based on string matching. *IEEE Trans. Inform. Theory*, 39(3):877–886, 1993.
- [16] T. Luczak and W. Szpankowski. A suboptimal lossy data compression based on approximate pattern matching. *IEEE Trans. Inform. Theory*, 43:1439–1451, 1997.
- [17] I. Kontoyiannis. An implementable lossy version of the Lempel-Ziv algorithm—part i: Optimality for memoeyless sources. *IEEE Trans. Inform. Theory*, 45(7):2293–2305, 1999.
- [18] M. Atallah, Y. Genin, and W. Szpankowski. Pattern matching image compression: Algorithmic and empirical results. In *Proc. Int. Conf. Image Processing, Lausanne, Switzerland*, volume II, pages 349–352, 1996.
- [19] J. C. Kieffer and E.-H. Yang. Sequential codes, lossless compression of individual sequences, and Kolmogorov complexity. *IEEE Trans. Inform. Theory*, 42(1):29–39, 1996.
- [20] M. Alzina, W. Szpankowski, and A. Grama. 2D-pattern matching image and video compression. In *Data Compression Conf.*, pages 424–433, 1999.
- [21] R. Zamir and K. Rose. Towards lossy Lempel-Ziv: Natural type selection. In *Proc. Inform. Theory Workshop, Haifa, Israel*, page 58, 1996.
- [22] R. Zamir and K. Rose. Natural type selection in adaptive lossy compression. *IEEE Trans. Inform. Theory*, 47(1):99–111, 2001.
- [23] Y. Kochman and R. Zamir. Adaptive parametric vector quantization by natural type selection. In *Data Compression Conference*, pages 392–401, 2002.
- [24] E.-H. Yang and Z. Zhang. Variable-rate trellis source encoding. *IEEE Trans. Inform. Theory*, 45(2):586–608, 1999.
- [25] D. S. Modha. Codelet Parsing: Quadratic-time, sequential, adaptive algorithms for lossy compression. In *Proc. DCC, Snowbird, UT*, March 24–27, 2003.
- [26] D. S. Modha. The art of making mistakes: A quadratic-time, sequential, adaptive algorithm for lossy compression. Technical Report RJ 10286, IBM Almaden Research Center, San Jose, CA, February 19, 2003.