A Lossy Data Compression Based on String Matching:
Preliminary Analysis and Suboptimal Algorithms

Tomasz Łuczak[1] and Wojciech Szpankowski[2]

[1] Mathematical Institute, Polish Academy of Science, 60-769 Poznań, Poland
[2] Dept. of Computer Science, Purdue University, W. Lafayette, IN 47907, USA

Abstract. A practical suboptimal algorithm (source coding) for lossy (non-faithful) data compression is discussed. This scheme is based on an approximate string matching, and it naturally extends lossless (faithful) Lempel-Ziv data compression scheme. The construction of the algorithm is based on a careful probabilistic analysis of an approximate string matching problem that is of its own interest. This extends Wyner-Ziv model to lossy environment. In this conference version, we consider only Bernoulli model (i.e., memoryless channel) but our results hold under much weaker probabilistic assumptions.

1 Introduction

Repeated patterns and related phenomena in words (sequences, strings) play a central role in many facets of telecommunications and theoretical computer science, notably in coding theory and data compression, in the theory of formal languages, and in the design and analysis of algorithms. For example: in faithful data compressions, such a repeated subsequence can be used to reduce the size of the original sequence (e.g., universal data compression schemes [15, 24, 22]); in exact string matching algorithms the longest suffix that matches a substring of the pattern string is used for "fast" shift of the pattern over a text string (cf. Knuth-Morris-Pratt and Boyer-Moore [1]; see also [8]).

However, in practice *approximate* repeated patterns are even more important. Non-faithful (or lossy) data compression and molecular sequence comparison are most notable examples. In this paper, we shall use approximate pattern matching to design suboptimal lossy (non-faithful) data compression. Hereafter, we shall think in terms of data compression, but most of our analysis and algorithms can be directly used to molecular sequences comparison (e.g., finding approximate palidroms).

We first briefly review some aspects of the distortion theory to put our results in the proper perspective. The reader is referred to [9, 10] for more details.

Consider a stationary and ergodic sequence $\{X_k\}_{k=-\infty}^{\infty}$ taking values in a finite alphabet $\mathcal{A}$. For simplicity of presentation, we consider only binary alphabet $\mathcal{A} = \{0,1\}$. We write X_m^n to denote $X_m X_{m+1} \dots X_n$.

In data compression, one investigates the following problem: Imagine a source of information generating a block $x_1^n = (x_1, \dots, x_n)$ which is a realization of the process X_1^n. To send it efficiently one codes it into another sequence $y_1 \dots y_\ell$ of length ℓ. Then, the compression factor is defined as $c(x_1^n) = \ell/n \le 1$ and its expected value is $C = Ec(X_1^n)$. What are the achievable values of C for lossless and lossy data compressions?

It is well known [9, 10, 13, 14, 24] that the average compression factor in a lossless data compression can asymptotically reach entropy rate, h. For the lossy transmission, one needs to introduce a measure of fidelity to find the achievable region of C. We restrict our discussion to Hamming distance defined as $d_n(x_1^n, \widetilde{x}_1^n) = n^{-1}\sum_{i=1}^n d_1(x_i, \widetilde{x}_i)$ where $d_1(x, \widetilde{x}) = 0$ for $x = \widetilde{x}$ and 1 otherwise $(x, \widetilde{x} \in \mathcal{A})$.

* This research was partially done while the authors were visiting INRIA in Rocquencourt, France. The authors wish to thank INRIA (project ALGO) for a generous support. Additional support for the first author was provided by KBN grant 2 1087 91 01, while for the second author by NSF Grants NCR-9206315 and CCR-9201078 and INT-8912631, and in part by NATO Collaborative Grant 0057/89.

Let now fix $D > 0$. Roughly speaking, a data compression (or a code) is called *non-faithful* or *lossy* or *D-faithful* if any set of sequences x_1^n lying within distance D of a representative sequence $\widetilde{x}_1^n$ is coded as $\widetilde{x}_1^n$.

The optimal compression factor depends on the so called rate-distortion function $R(D)$. This is defined as follows (we give the definition of the operational rate-distortion function): Let $B_D(w_n)$ be the set of all sequences of length n whose distance from the center w_n is smaller or equal to D, that is, $B_D(w_n) = \{x_1^n : \ d_n(x_1^n, w_n) \leq D\}$. We call the set $B_D(w_n)$ a D-ball. Consider now the set $\mathcal{A}^n$ of all sequences of length n, and let $\mathcal{S}_n$ be a subset of $\mathcal{A}^n$. We define $N(D, \mathcal{S}_n)$ as the *minimum* number of D-balls needed to cover $\mathcal{S}_n$. Then[1]

$$R_n(D, \varepsilon) = \min_{\mathcal{S}_n:\ P(\mathcal{S}_n) \geq 1-\varepsilon} \frac{\log N(D, \mathcal{S}_n)}{n} .$$

The operation rate-distortion is (cf. [13, 17])

$$R(D) = \lim_{\varepsilon \to 0} \lim_{n \to \infty} R_n(D, \varepsilon) . \tag{1}$$

Kieffer [13, 14] and Ornstein and Shields [17] proved that the compression factor in a D-faithful data compression is asymptotically equal to $R(D)$, and this cannot be improved. (Observe that $R(0) = h$.) Note, however, that to construct an optimal data compression one needs to "guess" the optimal (i.e., minimum) cover of the set $\mathcal{A}^n$ by D-balls. (This is actually identical of guessing a probability measure on $\mathcal{A}^n$ that minimizes the mutual information [9, 13, 19] which is equally difficult task!).

In this paper, we propose a practical suboptimal lossy data compression scheme that extends the Lempel-Ziv scheme and that achieves rate $r(D)$ which is asymptotically optimal for $D \to 0$, that is, $\lim_{D\to 0} r(D) = h$. Our scheme reduces to the following approximate pattern matching problem. Let the "database" sequence x_1^n be given. Find the longest L_n such that there exists $i \leq n$ in the database satisfying $d(x_i^{i-1+L_n}, x_{n+1}^{n+L_n}) \leq D$. We shall propose an algorithm that finds L_n on average in $O(n \cdot \mathit{poly}(\log n))$ steps (but in $O(n^2 \log n)$ steps in the worst case). More importantly, we also propose two compression schemes that are based on this algorithm and our probabilistic analysis. Actually, the real engine behind this study (and its algorithmic issues) is a probabilistic analysis of an approximate pattern matching problem.

Our probabilistic results are confined to the Bernoulli model (however, in the forthcoming journal version of the paper we extend them to mixing models; see Remark after Theorem 2 below). Thus, we assume that: *symbols from $\mathcal{A}$ are generated independently and "0" occurs with probability p while "1" with probability $q = 1 - p$.* We prove that $L_n/\log n \to 1/r(D)$ in probability (pr.) where $r(D)$ represents the rate distortion, and in general $r(D) \geq R(D)$, except the symmetric case ($p = q = 0.5$) in which $r(D) = R(D)$. But, we shall show that $\lim_{D\to 0} r(D) = \lim_{D\to 0} R(D) = h$. Surprisingly enough, $L_n/\log n$ does not converge almost surely (a.s.) but rather oscillates between two random variables $s_n/\log n$ and $H_n/\log n$ that converge almost surely to two *different* constants. This kind of behavior was already observed in the faithful case (cf. [20, 21]).

Our results extends those of Wyner and Ziv [22] and Szpankowski [20, 21] to lossy transmission. We observe, however, that in the lossless case the natural data structure around which practical schemes could be built is a suffix tree (cf. [20, 21]) or digital search tree. The situation with lossy data compression is much more complicated since the decoder at any time has as a database a sample of the *distorted* process. We shall propose two solutions to remedy this problem.

Our paper is close in spirit to the one of Steinberg and Gutman [19] who also considered a practical data compression scheme based on a string matching. But the authors of [19] studied the so called waiting time while we concentrate on approximate prefix analysis. Furthermore, the authors of [19] obtained only an upper bound while we establish here precise asymptotic results. Finally, we should mention that there are results (cf. [13, 17]) indicating the existence of the optimal (thus,

[1] All logarithms in this paper are with base 2 unless otherwise explicitly stated.

achieving the rate $R(D)$) data compression. However, these scheme are exponentially expensive in implementations (cf. [19]). Very recently, Zhang and Wei [23] proposed an asymptotically optimal lossy data compression that is based on the so called "gold washing" or "information-theoretical sieve" method.

2 Main Results

After formulating the pattern matching problem, we present some analytical (probabilistic) results. These results are of prime importance for the algorithmic issues which are discussed next.

2.1 Analytical Results

Let $\{X_k\}_{k=1}^{\infty}$ be a stationary ergodic sequence generated over a binary alphabet $\mathcal{A} = \{0, 1\}$. Wyner and Ziv [22] (see also [16, 20]) proposed the following mutation of the Lempel-Ziv data compression scheme: Assume first n symbols, X_1^n, are known to the transmitter and the receiver. Call it the *database* sequence. Find the longest prefix of X_{n+1}^{∞} that occurs at least once in the database. Say, this occurrence is at position $i_0 \leq n$ and it is of length $L_n - 1$. It was proved by Wyner and Ziv [22] that $L_n/\log n \to 1/h$ in probability (pr.), where h is the entropy of the alphabet. However, Szpankowski [20, 21] showed that $L_n/\log n$ does *not* converge almost surely (a.s.) to any constant but rather oscillates between two different constants.

Based on these results, Wyner and Ziv [22] proposed the following data compression scheme: The encoder sends the position i_0 in the database, the length $L_n - 1$ and one symbol, namely X_{n+L_n}. Using this information the decoder reconstructs the original message, and both the encoder and the decoder enlarge the database to the right, that is, the new database becomes $X_1^{n+L_n}$ or $X_{L_n}^{n+L_n}$ (the so called sliding window scheme). Based on the probabilistic results discussed above, one easily concludes that the compression ratio of such an algorithm is equal to the entropy, and it is asymptotically optimal. This scheme is called a faithful data compression scheme.

In this paper, we discuss a scheme that directly extends the above algorithm to a lossy data transmission with a fidelity criterion. As in the introduction, we define the Hamming distance $d(x_1^n, \tilde{x}_1^n)$ as the ratio of the number of mismatches between x_1^n and $\tilde{x}_1^n$ to the length n, and we assume that the database X_1^n is given (see Section 2.2 for a detailed discussion of this point). We construct the longest prefix of X_{n+1}^{∞} that is within distance $D > 0$ of a substring in the database. More precisely:

> Let L_n be the length of the largest prefix of X_{n+1}^{∞} such that there exists $i \leq n$ so that $d(X_i^{i-1+L_n}, X_{n+1}^{n+L_n}) \leq D$.

We call L_n the *depth* to mimic the name adopted in the faithful case (cf. [20, 21]).

As in the faithful case, the quality of compression depends on the probabilistic behavior of L_n. It turns out that its behavior depends on two other quantities, namely s_n and H_n defined in sequel.

The *height* H_n is the length of the longest substring in the database X_1^n for which there exists another substring in the database within distance D. More precisely: the height is equal to the largest K for which there exist $1 \leq i < j \leq n$ such that $d(X_i^{i-1+K}, X_j^{j-1+K}) \leq D$. In the proof, we shall need another definition of H_n which is presented below. Let $1(A)$ be the indicator function of the event A. Then, the following is true (this is a correct version of the definition presented in [4])

$$\{H_n \geq k\} = \bigcup_{l \geq k} \bigcup_{1 \leq i < j \leq n} \left\{ \sum_{t=1}^{l} 1(X_i^{i-1+t} \neq X_j^{j-1+t}) \leq Dl \right\} . \tag{2}$$

The *shortest path* s_n is defined as follows: Let $\mathcal{W}_k$ be the set of words of length k, and $w_k \in \mathcal{W}_k$. The shortest path s_n is the longest k such that for every $w_k \in \mathcal{W}_k$ there exists $1 \leq i \leq n$ such that $d(X_i^{i-1+k}, w_k) \leq D$.

Now, we are in a position to present our main results. As mentioned before, in this preliminary version we discuss only Bernoulli model in which "0" occurs with probability p and "1" with probability $q = 1 - p$. We also define $p_{\min} = \min\{p, q\}$ and $P = p^2 + q^2$. All proofs are delayed till the next section.

Theorem 1. *Let $h(D,x) = (1-D)\log((1-D)/x) + D\log(D/(1-x))$. Then*

$$\lim_{n\to\infty} \frac{s_n}{\log n} = \frac{1}{h(D, p_{min})} \qquad \text{(a.s.)} \ , \tag{3}$$

and

$$\lim_{n\to\infty} \frac{H_n}{\log n} = \frac{2}{h(D, P)} \qquad \text{(a.s.)} \ , \tag{4}$$

for $0 \le D < p_{\min}$. ■

Remark 1. Observe that for $D \ge p_{\min}$ the shortest path s_n and the height H_n with high probability grow faster than logarithmic function. However, this case is not too interesting from the algorithmic view point since the Hamming distance between the database sequence and a string consisting entirely either of zeros (when $p_{\min} = q$) or of ones (when $p_{\min} = p$) is smaller than D with high probability. Thus, we neglect this case in our analysis. □

The next theorem tells us about the probabilistic behavior of L_n which is really responsible for the asymptotic behavior of a lossy data compression scheme discussed below.

Theorem 2. *Let*

$$x_0 = \begin{cases} \frac{D}{2} + \frac{\sqrt{p^2q^2 + D^2(p-q)^2 - 2Dpq(p-q)^2} - pq}{2(p-q)} & \textit{for } p \ne q \\ \frac{D}{2} & \textit{for } p = q = 0.5 \ . \end{cases} \tag{5}$$

Define $r(D) = -\log F$ where

$$F = \frac{p^{2p-2x_0+D} q^{2q+2x_0-D}}{x_0^{x_0}(p-x_0)^{p-x_0}(D-x_0)^{D-x_0}(q-D+x_0)^{q-D+x_0}} \ . \tag{6}$$

Then, for any $\varepsilon > 0$ and large n

$$\Pr\left\{ \left| \frac{L_n}{\log n} - \frac{1}{r(D)} \right| \le \varepsilon \right\} \ge 1 - O\left(\frac{\log n}{n^{\varepsilon}}\right) \ , \tag{7}$$

that is, $L_n/\log n \to 1/r(D)$ (pr.). But, $L_n/\log n$ does not converge almost surely. More precisely,

$$\liminf_{n\to\infty} \frac{L_n}{\log n} = \frac{1}{h(D, p_{\min})} \qquad \text{(a.s.)} \qquad \limsup_{n\to\infty} \frac{L_n}{\log n} = \frac{2}{h(D, P)} \tag{8}$$

provided $0 \le D \le p_{\min}$. ■

Remark 2. Actually, in the journal version of this paper we will prove much stronger results that we announce in this remark. Consider a stationary, ergodic *mixing model* (cf. [7, 18, 20]) in which the sequence $\{X_k\}_{k=-\infty}^{\infty}$ is stationary and ergodic. In addition, it is mixing in strong sense, that is, (informally speaking) for two events A and B defined respectively of σ-algebra of $\{X_k\}_{-\infty}^{m}$ and $\{X_k\}_{m+b}^{\infty}$ for some integer b, the following holds

$$(1-\alpha(b))\Pr\{A\}\Pr\{B\} \le \Pr\{A \cap B\} \le (1+\alpha(b))\Pr\{A\}\Pr\{B\}$$

for some some $\alpha(b)$ such that $\lim_{b\to\infty}\alpha(b)=0$. Let now $P(B_D(w_n))$ and $P(B_D(X_1^n))$ denote the probabilities of all strings in the balls $B_D(w_n)$ and $B_D(X_1^n)$, respectively. Observe that $P(B_D(X_1^n))$ is in fact a random variable. Then, in the mixing model, we define three quantities as follows:

$$h_{\min}(D) = \lim_{n\to\infty} \frac{-\log\left(\min_{w_n}\{P(B_D(w_n))\}\right)}{n} \ , \tag{9}$$

$$h_1(D) = \lim_{n\to\infty} \frac{-\log\left(EP(B_D(X_1^n))\right)}{n} \ , \tag{10}$$

$$r(D) = \lim_{n\to\infty} \frac{-E\left(\log P(B_D(X_1^n))\right)}{n} \ . \tag{11}$$

It can be proved that the above limits exist in the mixing model. Observe also that in the Bernoulli model $h_{\min}(D)$ becomes $h(D,p_{min})$, and $h_1(D)$ is equivalent to $h(D,P)$. Also, $r(D)$ defined above can be computed as in Theorem 2 for the Bernoulli model.

Having these definitions in mind, we can now formulate our general results. First of all, we shall prove in the forthcoming paper that

$$\lim_{n\to\infty}\frac{s_n}{\log n} = \frac{1}{h_{\min}(D)} \qquad \text{(a.s.)} \ ,$$

provided $\alpha(b)$ decays to zero faster than any polynomial, that is, for all $m \geq 0$ we have $b^m\alpha(b) \to 0$. Secondly, the height H_n becomes

$$\lim_{n\to\infty}\frac{H_n}{\log n} = \frac{2}{h_1(D)} \qquad \text{(a.s.)} \ ,$$

provided $\alpha^2(b)$ is summable, that is $\sum_{b=0}^{\infty}\alpha^2(b) < \infty$. Finally, Theorem 2 generalizes to the following

$$\lim_{n\to\infty}\frac{L_n}{\log n} = \frac{1}{r(D)} \qquad \text{(pr.)} \ ,$$

if $\alpha(d) \to 0$. But, as in Theorem 2, L_n does not converge almost surely, and only the following can be claimed

$$\liminf_{n\to\infty}\frac{L_n}{\log n} = \frac{1}{h_{\min}(D)} \quad \text{(a.s.)} \qquad \limsup_{n\to\infty}\frac{L_n}{\log n} = \frac{2}{h_1(D)}$$

under the same condition on $\alpha(b)$ as for s_n. $\Box$

A lossy data compression scheme based on Theorem 2 is presented below. Observe that such a scheme is more intricate than in the lossless case due to the fact that the decoder and encoder have different database (i.e., the decoder has as a database a sample of the distorted process). Before we discuss algorithmic issues concerning such schemes, we first estimate the compression factor.

It is rather clear that any compression scheme based on Theorem 2 should have compression factor C equal to $r(D)$. Indeed, we observe that, as in the faithful case, any non-faithful data compression scheme based on the approximate string matching needs a pointer to the database and the length of the approximate matching. The former information costs $\log n$ while the latter can be decoded in $O(\log\log n)$ bits. In other words, instead of sending $(1/r(D))\cdot\log n$ bits of L_n, one transmits $\log n + O(\log\log n)$ bits, thus the compression factor is $r(D)$.

In view of the above, one may ask how close is the rate (compression factor) $r(D)$ of our scheme to the optimal compression factor equal to $R(D)$ as defined in (1). An explicit formula for $R(D)$ seems to be unknown except for the Bernoulli case. In this case [9], $R(D) = h - h(D)$ where $h = -p\log p - q\log q$ is the entropy of the memoryless channel, and $h(D) = -D\log D - (1-D)\log(1-D)$. Note that $R(0) = h$. From Theorem 2 formulæ (5)-(6) we conclude that the scheme is:

– asymptotically optimal in the limiting case, namely

$$\lim_{D\to 0} R(D) = \lim_{D\to 0} r(D) = h \ , \tag{12}$$

– asymptotically optimal in the symmetric Bernoulli case ($p = q = 0.5$) since

$$r(D) = R(D) = \log 2 - h(D) \ . \tag{13}$$

In general, $r(D) \geq R(D)$, however, a numerical study shows that the discrepancy between $R(D)$ and $r(D)$ is not too big as one may conclude from Figure 1 which presents the gains in compression factors, namely, $h/r(D)$ and $h/R(D)$ versus D.

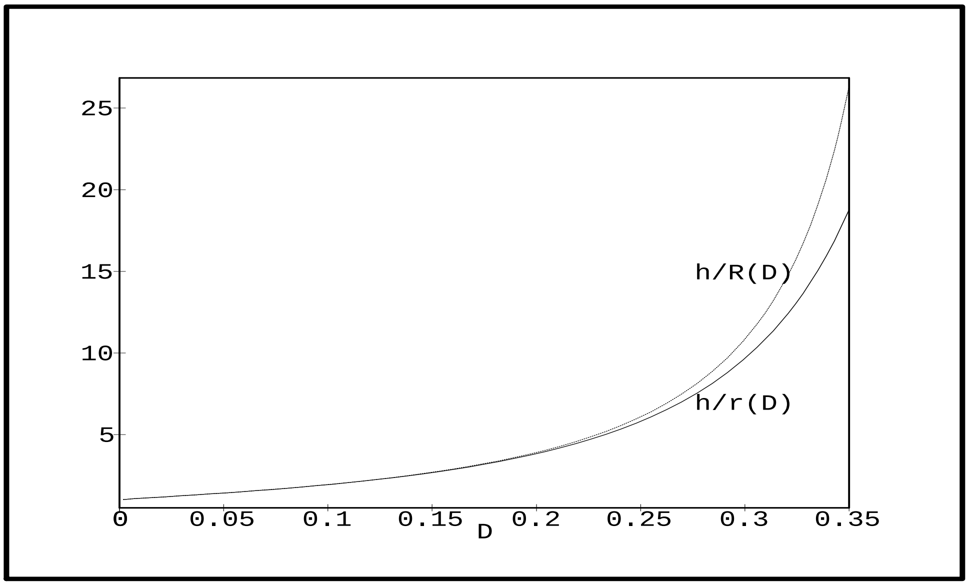

Fig. 1. Comparison of gains in the compression factors for $p = 0.4$

2.2 Algorithmic Results

As mentioned above, the non-faithful data compression is much more intricate than the faithful one due to two reasons: In the faithful case, the prefix of length L_n can be found in $O(n)$ time-complexity by a simple application of the suffix tree structure (cf. [20]). Secondly, the encoder and the decoder have different view on the database. These two problems must be solved in order to obtain an efficient lossy data compression based on Theorem 2, and we discuss them in sequel.

We start with an approximate pattern matching algorithm that finds the longest prefix of X_{n+1}^{∞} that is within distance D of a substring in the database. We shall write below lowercase letter x_m^n to denote a realization of the process X_m^n.

The following algorithm is an adaptation of the idea already applied in Atallah *et al.* [3] to another problem.

Algorithm `PREFIX`

begin
  For $i = n$ **to** 1 **do**
    Apply Fast Fourier Transform (FFT) to compute matches between x_{n+1}^{n+i} and $\{x_j^{j-1+i}\}_{j=1}^{n}$,
    Select $j \leq n$ that gives the longest substring with $(1 - D)\%$ of matches,
  doend
end

Clearly this algorithm works in $O(n^2 \log n)$ time-complexity since the FFT needs $O(n \log n)$ to compute matches between a string and *all* substrings of another string.

Although, $O(n^2 \log n)$ algorithm sounds like a good solution, it is too expensive in most applications when `PREFIX` is expected to be run very often. One needs an algorithm that *most* of the time is linear or poly-linear. Below, we discuss two possible solutions.

The problem with the previous `PREFIX` algorithm is the **do-loop** which requires n iterations. One possible solution (suggested by M. Atallah) is to apply binary search. The idea of the new algorithm `PREFIX-BS` is as follows. Let $Y_1^n = X_{n+1}^{2n}$. Using FFT we check if Y_1^n has $(1-D)\%$ of matches with any substrings of X_1^n. If the answer is YES we stop, otherwise we continue the binary search. That is, we divide the substring Y_1^n into two halves, and check whether $Y_1^{n/2}$ approximately occurs (i.e., with less than $D\%$ mismatches) in X_1^n. Again if answer is YES, we are fine and start investigating $Y_1^{3n/4}$. The only problem arises, however, when the algorithm returns NO. Say, it happens when checking $Y_1^{n/2}$. This, unfortunately does not mean – as in the classical binary search – that we can proceed to $Y_1^{n/4}$ since still there is a possibility that $Y_1^{3n/4}$ almost occur in X_1^n. There are two possibilities:

(A) We use a heuristic `PREFIX-BSH` that searches for YES in the right-hand side of the string Y. More precisely, if NO occured when investigating $Y_1^{n/2}$ before we consider $Y_1^{n/4}$ we check only few, say two, up-searches to see if YES does occur. For example, for NO at position $Y_1^{n/2}$ we only investigate $Y_1^{3n/4}$ and/or $Y_1^{5n/8}$ for the approximate pattern matching. If in any case, we receive the answer YES, we continue exact binary search. Otherwise, we abandon the up-search, and the next check is at $Y_1^{n/4}$.

(B) We append the binary search with the exact search to obtain the following algorithm that is further called `PREFIX-BSE`. As before, consider NO at $Y_1^{n/2}$. Then, we search all prefixes $Y_1^{n/2+i}$ with $i = 1, \ldots, n/2$ until YES is obtained. If no YES occured during such a search, we then move to $Y_1^{n/4}$ as discussed above.

Our heuristic `PREFIX-BSH` works in $O(n \log n)$ steps in the worst case, but it returns the true value only with high probability (whp) and sometimes we might be off. On the other hand, the algorithm `PREFIX-BSE` always returns the longest prefix, but it is slower than the previous one, that is, its complexity is $O(n^2 \log n)$ in the worst case. On average, however, this algorithm is $O(n \cdot poly(\log n))$. In our experimental studies, we used `PREFIX-BSE`.

To complete the description of our lossy data compression scheme we must describe how the database is updated. There are two options, too. In the first one, the database is sent faithfully by the encoder, for example using the Lempel-Ziv scheme. The lossy compression refers now only to the new transmissions and the references are made to the common copy of the database. We also systematically measure the compression ratio, and once it falls below some specified level, a new faithful transmission of database is required. This procedure might be on-line.

The above scheme seems to be appropriate for situations when the database is kept unchanged for some time. For example, when sending pictures from a satellite, usually several pictures have the same background, hence the same database, so clearly our scheme is suitable for such transmission.

In the case when the database is varying quickly, another algorithm is needed. We suggest the following one. Instead of sending lossly a faithful database, we rather send faithfully (e.g., by Lempel-Ziv scheme) a non-faithful (distorted) database that is maintained simultaneously by the encoder and decoder.

We only briefly present the main idea of this scheme leaving details to a journal version (cf. [19]). When a new prefix of length L_n of X_{n+1}^{∞} is constructed, it is *not* added directly to the database but rather we add the center $w_1^{L_n}$ of a ball $B_D(w_1^{L_n})$ to which the prefix falls. For example, this can be accomplished by finding the prefix of length L_n by approximate pattern matching, say `PREFIX-BSE`, in the distorted database $\widetilde{X}_1^n$ that stores only the *centers* of balls $B_D(\cdot)$. Then, the encoder transmits faithfully the distorted version of the database $\widetilde{X}_1^n$ (i.e., the centers of D-balls).

More precisely, the encoder sends only the pointer to the distorted database (maintained the same by the encoder and decoder) and the length L_n. Since the pointer costs $\log n$ and by Theorem 2 we have $L_n \sim 1/r(D) \log n$, so one can conclude that the compression factor is still asymptotically equal to $r(D)$.

3 Probabilistic Analysis

In this section, we present a sketch of proofs for Theorems 1 and 2. To simplify our analysis, we observe that the following formulation of the problem turns out to be asymptotically equivalent to our original model (see [20, 21] where a similar approach is used).

Let us generate unbounded sequences $X(1), X(2), \ldots, X(m+1)$ according to the original distribution, *independently* from each other. Let $\hat{L}_m$ denote the length of the longest prefix of $X(m+1)$ that lies within the distance D from the prefix of $X(i)$ for some $i = 1, 2, \ldots, m$. Let also $\hat{H}_m$ be the largest k such that

$$d(X_1^k(i), X_1^k(j)) < D \quad \text{for some} \quad i, j,\ 1 \le i < j \le m \ .$$

Finally, let $\hat{s}_m$ denote the length of the shortest string that has no approximate match among prefixes of $X(1), X(2), \ldots, X(m)$.

One can show (cf. [20, 21]) that the behaviour of random variables $\hat{L}_m$, $\hat{H}_m$ and $\hat{s}_m$ defined for the above *independent model* asymptotically resembles that of L_m, H_m and s_m in the original model provided $m = O(n/\log n)$. Thus, throughout the following section we shall work within the framework of the independent model, and we do not distinguish between these two cases writing L_m, H_m and s_m instead of $\hat{L}_m$, $\hat{H}_m$ and $\hat{s}_m$.

3.1 The Shortest Length

We first prove Theorem 1 for s_n, that is, (3). Let us introduce some additional notation. To recall, we define $\mathcal{W}_k$ as the set of words of length k. For a $w_k \in \mathcal{W}_k$ we write $P(w_k)$ for the probability of w_k. Let $w_{\min} \in \mathcal{W}_k$ be such that $P(w_{\min}) = \min_{w \in \mathcal{W}_k}\{P(w)\}$. We also write $P(B_D(w_{\min}))$ as the probability of a D-ball centered at $w_{\min}$. It is easy to verify that $P(B_D(w_{\min})) = \min_{w_k \in \mathcal{W}_k} \Pr\{B_D(w_k)\}$.

As defined before, the shortest path s_m is the longest k such that for every $w_k \in \mathcal{W}_k$ there exists $1 \le i \le m$ such that $d(X_1^k(i), w_k) \le D$. Clearly, the following is true

$$\Pr\{s_m > k\} \le m \min_{\mathcal{W}_k} \Pr\{d(X_1^k(i), w_k) \le D\} = mP(B_D(w_{\min})) \ . \tag{14}$$

To estimate the above probability, one needs to assess $P(B_D(w_{\min}))$. Let $p_{\min} = \min\{p, q\}$. We note that $w_{\min}$ is a string that consists of all zeros or all ones depending whether $p < q$ or $p > q$, hence

$$P(B_D(w_{\min})) = \sum_{j=0}^{kD} \binom{k}{j} p_{\min}^{k-j} (1 - p_{\min})^j \ .$$

By Stirling's formula we have

$$\binom{k}{kD} \sim \left(\frac{1}{((1-D)^{1-D} D^D)}\right)^k \ .$$

Thus, for large k and $D \le p_{\min}$

$$\left(\left(\frac{p_{\min}}{1-D}\right)^{1-D} \left(\frac{1-p_{\min}}{D}\right)^D\right)^k \le P(B_D(w_{\min})) \le k \left(\left(\frac{p_{\min}}{1-D}\right)^{1-D} \left(\frac{1-p_{\min}}{D}\right)^D\right)^k \ .$$

In view of the above, we obtain $P(B_D(w_{\min})) \sim 2^{-kh(D,p_{\min})}$ where $h(D,p_{\min}) = (1-D)\log((1-D)/p_{\min}) + D\log(D/(1-p_{\min}))$. Thus, for $k = \lfloor (1+\varepsilon)h^{-1}(D,p_{\min})\log m \rfloor$ we conclude that $\Pr\{s_m > (1+\varepsilon)h^{-1}(D,p_{\min})\log m\} \le 1/m^{\varepsilon}$, which proves the upper bound for the convergence in probability of s_m.

To get the lower bound for s_m, we proceed as follows. Note that

$$\Pr\{s_m < k\} \le \sum_{\mathcal{W}_k} (1 - P(B_D(w_k)))^m \le 2^k (1 - P(B_D(w_{\min})))^m \ .$$

Using the above estimate for $P(B_D(w_{\min}))$ and setting $k = \lfloor (1-\varepsilon)h^{-1}(D,p_{\min})\log m \rfloor$ we finally obtain

$$\Pr\{s_m < k\} \le \exp(-m^{\varepsilon/2})$$

which is the desired lower bound.

From the above, we conclude that $s_m/\log m \to 1/h(D,p_{\min})$ (pr.) but the rate of convergence (upper bound) does not yet warrant direct application of the Borel-Cantelli Lemma. Nevertheless, one can use Kingman's idea as in [18, 20, 21] to extend this result to the almost sure convergence. Indeed, one selects a subsequence like $m_r = s2^r$ along which $s_{m_r}/\log m$ converge almost surely (a.s.), and then by noting that s_m is a nondecreasing sequence with respect to m one can extend the last assertion to all m. This completes the proof for s_m, and actually for s_n since $m = O(n/\log n)$, hence all the results above easily extend to this case, too.

3.2 The Height

The height was already treated by Arratia and Waterman [4] (cf. Theorem 1 in [4]) for the independent model, and the string model can be analyzed along the same lines.

For completeness, we only present the derivation of the upper bound, which also corrects a minor problem of [4]. From the definition (2) we have

$$\begin{aligned}\{H_m \ge k\} &= \bigcup_{l\ge k} \bigcup_{1\le i<j\le m} \left\{ \sum_{t=1}^{l} 1\{X_t(i) = X_t(j)\} \ge al \right\} \\ &= \bigcup_{l\ge k} \bigcup_{1\le i<j\le m} \{d(X(i),X(j)) \le D\} \ .\end{aligned}$$

(In [4] the first union symbol was missing.)

Now, we consider $M = m(m-1)/2$ new sequences $Y(1), \ldots Y(M)$ such that $Y_k(t) = 1$ ($t = 1, \ldots, M$, $k = 1, \ldots$) if and only if for $1 \le i,j \le m$ resulting in t there is a match between $X_k(i)$ and $X_k(j)$, i.e., $X_k(i) = X_k(j)$; otherwise $Y_k(t) = 0$. Note that $\Pr\{Y_k(t) = 1\} = P = p^2 + q^2$.

The rest is easy, and we obtain

$$\Pr\{H_m \ge k\} \le m^2 \sum_{l\ge k} P(B_D(Y(1),w_l)) \ ,$$

for some $w_l \in \mathcal{W}_l$. From our previous estimate of the probability of a D-ball, we observe that $P(B_D(Y(1),w_l) \sim 2^{-lh(D,P)}$. Thus, for $k = \lfloor (1+\varepsilon)h^{-1}(D,P)\log m \rfloor$ and a constant B

$$\Pr\{H_m \ge k\} \le Bn^2 2^{-kh(D,P)} = B/m^{2\varepsilon}$$

which is the desired upper bound. The lower bound can be derived by using the "second moment method" in a similar fashion as in [4] (cf. also [20, 21]).

So, far only convergence in probability was derived. But using again the Kingman trick, and noting that H_m is nondecreasing, we prove Theorem 1.

3.3 The Depth

Now, we prove Theorem 2, and we begin with the convergence in probability, that is, we establish (7). To accomplish our task, we need to show that a prefix of an independently generated string $X(m+1)$ of length L_m is within distance D of $X(i)$ for some $1 \le i \le m$, that is, $d(X_1^{L_m}(m+1), X_1^{L_m}(i)) \le D$. We prove that $L_m/\log m \to 1/r(D)$ (pr.) where $r(D)$ is defined in Theorem 2.

Let w_k be a *given* and *typical* word of length k. More precisely, $w_k \in \mathcal{W}_k$ and by Shannon-McMillan-Breiman Theorem (cf. [7, 9, 10]) $P(w_k) \sim 2^{-kh}$ where h is the entropy of the alphabet. In the above $P(w_k)$ has the meaning of probability of w_k occurrence, that is, $P(w_k) = p^{|0|_w} q^{|1|_w}$ where $|0|_w$ ($|1|_w$) denotes the number of zeros (ones) in w_k. For the Bernoulli model, we can say that with high probability the number of "0" and "1" in w_k is approximately equal to $kp \mp j$ and $kq \mp j$ where $j = o(k)$, respectively. Below, to simplify further discussion we assume that these numbers are $\lfloor kp \rfloor$ and $\lfloor kq \rfloor$ respectively (and actually we ignore the floor function). Naturally, $-\log P(w_k) \sim kh$.

We should stress that the word w_k is deterministic, but since it is also typical, the prefix of $X(m+1)$ of length k is close in probability to w_k. More specifically, for any $\varepsilon > 0$

$$\lim_{k\to\infty} \Pr\{|k^{-1}\log P(X_1^k(m+1)) - k^{-1}\log P(w_k)| \ge \varepsilon\} = 0 \ . \tag{15}$$

The above implies that instead of working with random string $X(m+1)$ we can work with deterministic word w_k provided the bounds on L_m hold uniformly for all w_k.

Let now Z_k be a random variable denoting the number of strings $X_1^k(1), \ldots, X_1^k(m)$ that lie within distance D from w_k, that is $Z_k = |\{1 \le i \le m;\ d(X_1^k(i), w_k) \le D\}|$. Due to our deterministic choice, Z_k has the binomial distribution with parameter m and $P_k \equiv P(B_D(w_k))$, i.e.,

$$\Pr\{Z_k = \ell\} = \binom{m}{l} P_k^\ell (1-P_k)^{k-\ell} \ .$$

The rest is a simple application of the *first moment method* and the *second moment method* (cf. [2]). Indeed,

$$\Pr\{Z_k = 0\} = \Pr\{D_n < k\} \le \frac{var\ Z_k}{(EZ_k)^2} = \frac{1-P_k}{mP_k} \tag{16}$$

$$\Pr\{Z_k > 0\} = \Pr\{D_n \ge k\} \le EZ_k = mP_k \ . \tag{17}$$

To complete the proof, we need to estimate the probability P_k which is discussed next.

Clearly, the following is true (for $P(w_k) = p^{\lfloor kp \rfloor} q^{\lfloor kq \rfloor}$)

$$P_k = P(B_D(w_k)) = \sum_{0 \le l+r \le kD} \binom{kp}{l}\binom{kq}{r} p^{kp+l-r} q^{kq-l+r}$$

where we assumed above for simplicity that kp and kq are integers. Let now $x = l/k$, and define

$$P_x = \binom{kp}{xk}\binom{kq}{(D-x)k} p^{k(p-x)+(D-x)k} q^{kx+k(q-D+x)} \ . \tag{18}$$

From the above, we immediately observe that

$$C \max_x\{P_x\} \le P(B(w_k)) \le Ck^2 \max_x\{P_x\}$$

where C is a constant. Thus, by the above $\log P(B_D(w_k)) \sim \log(\max_x\{P_x\})$, and it suffices to compute $\max_x\{P_x\}$.

Observe that by Stirling's formula

$$\binom{kp}{xk} \sim \left(\frac{p^p}{x^x(p-x)^{p-x}}\right)^k \ .$$

Thus, $P_x \sim (F(x))^k$ where

$$F(x) = \frac{p^{2p-2x+D} q^{2q+2x-D}}{x^x (p-x)^{p-x} (D-x)^{D-x} (q-D+x)^{q-D+x}} \ .$$

We need the following restriction on x: $\min\{0, p-q\} \le x \le \max\{p, D\}$.

Finally, to maximize $F(x)$ with respect to x, we are looking for x_0 such that $F'(x_0) = 0$. It turns out that this x_0 must solve the following quadratic equation

$$x^2(p-q) + x(pq + D(q-p)) - pq^2 = 0 \ .$$

The solution x_0 of the above is given by (5) in Theorem 2.

In summary, we have just proved that $P(B_D(w_k) \sim 2^{-kr(D)}$. Thus, by (15) and (16) with $k = \lfloor (1-\varepsilon)\frac{\log m}{r(D)} \rfloor$ we obtain the lower bound, while by (15) and (17) with $k = \lfloor (1+\varepsilon)\frac{\log m}{r(D)} \rfloor$ we derive the upper bound, which complete the proof of the convergence in probability of L_m.

To establish the second part of Theorem 2, namely (8), we proceed along the lines of [18, 20, 21]. More specifically, we note that $s_n \le L_n \le H_n$, and infinitely often (i.o.) $L_n = s_n$ as well as $L_n = H_n$. This, and Theorem 1, suffice to derive (8).

References

1. A.V. Aho, Algorithms for Finding Patterns in Strings, in *Handbook of Theoretical Computer Science. Volume A: Algorithms and Complexity* (ed. J. van Leeuwen), 255-300, The MIT Press, Cambridge (1990).
2. N. Alon and J. Spencer, *The Probabilistic Method*, John Wiley&Sons, New York (1992).
3. M. Atallah, P. Jacquet and W. Szpankowski, Pattern matching with mismatches: A probabilistic analysis and a randomized algorithm, *Proc. Combinatorial Pattern Matching*, Tucson, Lecture Notes in Computer Science, 644, (eds. A. Apostolico, M. Crochemore, Z. Galil, U. Manber), pp. 27-40, Springer-Verlag 1992.
4. R. Arratia and M. Waterman, The Erdös-Rényi Strong Law for Pattern Matching with Given Proportion of Mismatches, *Annals of Probability*, 17, 1152-1169 (1989).
5. R. Arratia, L. Gordon, and M. Waterman, The Erdös-Rényi Law in Distribution for Coin Tossing and Sequence Matching, *Annals of Statistics*, 18, 539-570 (1990)
6. T. Berger, *Rate Distortion Theory: A Mathematical Basis for Data Compression*, Englewood Cliffs, NJ: Prentice-Hall, 1971.
7. P. Billingsley, *Convergence of Probability Measure*, John Wiley & Sons, New York, 1968.
8. W. Chang, and E. Lawler, Approximate String Matching in Sublinear Expected Time, *Proc. of 1990 FOCS*, 116-124 (1990).
9. T.M. Cover and J.A. Thomas, *Elements of Information Theory*, John Wiley&Sons, New York (1991).
10. I. Csiszár and J. Körner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*, Academic Press, New York (1981).
11. J. Feldman, r-Entropy, Equipartition, and Ornstein's Isomorphism Theory in R^n, *Israel J. Math.*, 36, 321-345 (1980).
12. P. Jacquet and W. Szpankowski, Autocorrelation on Words and Its Applications. Analysis of Suffix Tree by String-Ruler Approach, *J. Combinatorial Theory. Ser. A*, (1994); to appear.
13. J.C. Kieffer, Strong Converses in Source Coding Relative to a Fidelity Criterion, *IEEE Trans. Information Theory*, 37, 257-262 (1991).
14. J. C. Kieffer, Sample Converses in Source Coding Theory, *IEEE Trans. Information Theory*, 37, 263-268 (1991).
15. A. Lempel and J. Ziv, On the Complexity of Finite Sequences, *IEEE Information Theory* 22, 1, 75-81 (1976).
16. D. Ornstein and B. Weiss, Entropy and Data Compression Schemes, *IEEE Information Theory*, 39, 78-83 (1993).
17. D. Ornstein and P. Shields, Universal Almost Sure Data Compression, *Annals of Probability*, 18, 441-452 (1990).
18. B. Pittel, Asymptotic Growth of a Class of random Trees, *Annals of Probability*, 13, 414 - 427 (1985).

19. Y. Steinberg and M. Gutman, An Algorithm for Source Coding Subject to a Fidelity Criterion, Based on String Matching, *IEEE Trans. Information Theory*, 39, 877-886 (1993).
20. W. Szpankowski, Asymptotic Properties of Data Compression and Suffix Trees, *IEEE Trans. Information Theory*, 39, 1647-1659 (1993).
21. W. Szpankowski, A Generalized Suffix Tree and Its (Un)Expected Asymptotic Behaviors, *SIAM J. Computing*, 22, 1176-1198 (1993).
22. A. Wyner and J. Ziv, Some Asymptotic Properties of the Entropy of a Stationary Ergodic Data Source with Applications to Data Compression, *IEEE Trans. Information Theory*, 35, 1250-1258 (1989).
23. Z. Zhang and V. Wei, An On-Line Universal Lossy Data Compression Algorithm via Continuous Codebook Refinement, submitted to a journal.
24. J. Ziv and A. Lempel, A Universal Algorithm for Sequential Data Compression, *IEEE Trans. Information Theory*, 23, 3, 337-343 (1977).