Available at
www.ElsevierMathematics.com
POWERED BY SCIENCE @ DIRECT®
Discrete Applied Mathematics 135 (2004) 245–254
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

Redundancy estimates for the Lempel–Ziv algorithm of data compression☆,☆☆

V.N. Potapov
Sobolev Institute of Mathematics, SB RAS, pr. Ak. Koptyuga 4, 630090 Novosibirsk, Russia

Abstract

The problem of non-distorting compression (or coding) of sequences of symbols is considered. For sequences of asymptotically zero empirical entropy, a modification of the Lempel–Ziv coding rule is offered whose coding cost is at most a finite number of times worse than the optimum. A combinatorial proof is offered for the well-known redundancy estimate of the Lempel–Ziv coding algorithm for sequences having a positive entropy. © 2002 Elsevier B.V. All rights reserved.

Keywords: Data compression; Lempel–Ziv algorithm

0. Introduction

Lempel and Ziv [18,19] offered methods of coding (hereafter called LZ77 and LZ78) which were widely applied to the data compression problem since then. Nowadays, a lot of modifications of these ideas are known [1,4,5,9,13,14,17]. Using algorithms based on Lempel–Ziv type rules in software development pushes interest to theoretical bounds on the quality of compression provided by these rules. In recent years, asymptotic estimates were obtained for the coding redundancy of various modifications of the Lempel–Ziv algorithm [8,10–12,15,16]. The estimate of special importance for practice is that of empirical redundancy $R(f, x_1^n)$ of a coding f of a sequence x_1^n consisting of n symbols from a finite alphabet. The value $R(f, x_1^n)$ is defined as the difference between the length of the code $f(x_1^n)$ of x_1^n and the empirical entropy $H(x_1^n)$ of this sequence, where the length of the code and the entropy are scaled per symbol of the sequence.

☆ Translated from Discrete Anal. Oper. Res. 6 (2) (1999) 70–81.
☆☆ Supported by RFBR (Grant 99-01-00531) and Federal Aim Program "Integration" (Grant 473).

PII: S0166-218X(02)00308-6

The best known redundancy estimates for the Lempel–Ziv method belong to Savari [11,12]:

$$R(f_1,x_1^n) = \mathrm{O}\left(\frac{1}{\log n}\right) \tag{1}$$

and

$$R(f_2,x_1^n) \leqslant \frac{CH(x_1^n)\log\log n}{\log n}(1+\mathrm{o}(1)) \tag{2}$$

as $n \to \infty$ and $\lim_{n\to\infty} H(x_1^n) > 0$, where $C=2$ and f_1 and f_2 are obtained according to LZ78 and LZ77, respectively. By log we mean the logarithm to the base 2. However, there exist examples of non-periodic sequences x (x_1^n being the prefix of x of length n) such that

$$\lim_{n\to\infty} \frac{R(f,x_1^n)}{H(x_1^n)} = \infty,$$

where the coding f is obtained by LZ77 or LZ78.

In the present paper, we offer a coding rule combining the algorithms LZ77 and LZ78. For a coding f built according to this rule, the redundancy estimate (2) holds for $C=1$ if $\lim_{n\to\infty} H(x_1^n) > 0$, and the redundancy estimate

$$R(f,x_1^n) = \mathrm{O}(H(x_1^n))$$

holds for an arbitrary non-periodic x. So, unlike LZ77 and LZ78, the algorithm offered guarantees the code of a sequence to be at most a finite number of times longer than its empirical entropy.

Besides, we offer a direct combinatorial proof of (2) with $C=1$ for LZ78 and $C=3$ for LZ77. The estimates obtained are somewhat worse than the known estimates (1) and (2) resulting from more cumbersome methods of probability theory.

1. Basic definitions

Let $A=\{a_1,\dots,a_{|A|}\}$ be a finite alphabet, and $A^* = \bigcup_{n=1}^{\infty} A^n$ be the set of all finite sequences of letters of A. Given words x, y, we denote their concatenation by xy. The word consisting of letters of a word $x = a_{i_1}\dots a_{i_n}$ starting with the lth letter and ending by rth one will be denoted by x_l^r; so, $x_l^r = a_{i_l}\dots a_{i_r}$. The word consisting of letters of x following a subword y will be denoted by $x(y) = a_{i_1}\dots a_{i_m}$; so, for each letter a_{i_j}, $1 \leqslant j \leqslant m$, occurring in $x(y)$ there exist words x_1^l and x_r^n such that $x = x_1^l y a_{i_j} x_r^n$. The length of x will be denoted by $|x|$.

The empirical entropy (of order 0) of a word $x_1^n \in A^n$ (see [6]) is

$$H(x_1^n) = \sum_{i=1}^{|A|} \frac{r_i}{n}\log\frac{n}{r_i}, \tag{3}$$

where r_i is the number of occurrences of a_i to x_1^n. Using the Stirling formula and (3), we obtain

$$H(x_1^n) = \frac{1}{n}\log\frac{n!}{r_1!r_2!\dots r_{|A|}!} + \alpha(x_1^n), \tag{4}$$

where $\alpha(x_1^n) \geqslant 0$ and $\alpha(x_1^n) \leqslant \alpha'(n) \to 0$ as $n \to \infty$.

The empirical entropy of order k of a word $x_1^n \in A^n$ (see [3]) is the value

$$H_k(x_1^n) = \sum_{y\in A^k} \frac{n(y)}{n} H(x_1^n(y)), \tag{5}$$

where $n(y) = |x_1^n(y)|$.

A coding is an injection $f : A^* \to E^*$ ($E = \{0,1\}$) taking each word on A to a binary sequence, the code of this word. A coding f is called prefix if for any distinct words x_1^n and y_1^n of length n on A the code $f(x_1^n)$ is not equal to a prefix of $f(y_1^n)$.

In what follows, we shall use the prefix code $\gamma(n)$ for positive integers offered by Elias [2]. (For similar earlier codes, see [7].) For each positive integer n, we have

$$|\gamma(n)| = 2\lfloor\log(\lfloor\log n\rfloor + 1)\rfloor + \lfloor\log n\rfloor + 1. \tag{6}$$

The (empirical) redundancy of order k of a coding f for a word x_1^n is

$$R_k(f, x_1^n) = \frac{1}{n}|f(x_1^n)| - H_k(f, x_1^n). \tag{7}$$

Let us consider the set $X(x_1^n) \subset A^n$ consisting of words in which the number of occurrences of a_i, $\leqslant i \leqslant |A|$, is the same as in x_1^n. Then for every prefix coding f and $x \in A^\infty$ the equalities (4) and (7) imply

$$\lim_{n\to\infty}\left(\sup_{z\in X(x_1^n)} R(f,z)\right) \geqslant 0.$$

Analogously, it can be shown that

$$\lim_{n\to\infty}\left(\sup_{z\in X_k(x_1^n)} R_k(f,z)\right) \geqslant 0,$$

where $X_k(x_1^n) \subset A^n$ is the set of $z \in A^n$ such that $z(y)$ and $x_1^n(y)$ have the same set of frequencies of letters for all $y \in A^k$.

2. Lempel–Ziv coding rule and its modifications

The algorithm LZ77 [18] consists in dividing the word $x_1^n \in A^n$ to be coded to subwords σ_i, $1 \leqslant i \leqslant m$, as follows. Let a prefix of x_1^n be already divided, i. e., let it be equal to the concatenation of subwords $\sigma_1, \sigma_2, \dots, \sigma_i$ and x_1^n be equal to $\sigma_1 \dots \sigma_i x_{l_i}^n$. We choose the longest sequence $x_{l_i}^{r_i}$ which already occurred in the prefix $x_1^{r_i-1}$ of x_1^n, i. e., $x_{l_i}^{r_i} = x_{l_i-n_i}^{r_i-n_i}$, where $1 \leqslant n_i < l_i$. Define the next subword σ_{i+1} as $\sigma_{i+1} = x_{l_i}^{r_i} a_{p_i}$, where a_{p_i} is the letter of x_1^n following $x_{l_i}^{r_i}$. The code of each subword

σ_{i+1} is the triplet $(r_i - l_i, n_i, p_i)$. For example, the sequence $a_2a_1a_2a_1a_1a_2a_1a_1a_2a_2$ divides to subwords $a_2, a_1, a_2a_1a_1, a_2a_1a_1a_2a_2$ and is coded by the sequence of triplets $(0,0,2),(0,0,1),(2,2,1),(4,3,2)$. We write the first number in a triplet $(r_i - l_i, n_i, p_i)$ by the coding γ, and the second and third ones as binary numbers of length $\lfloor \log n \rfloor + 1$ bits and $\lfloor \log |A| \rfloor + 1$ bits, respectively. Then due to (6) we have

$$|f_1(x_1^n)| \leqslant \sum_{i=1}^{m} (\log n + \log|\sigma_i| + 2\log(1 + \log|\sigma_i|) + \log|A| + 3), \tag{8}$$

where the coding f_1 is built by the rule LZ77, and m is the number of subwords σ_i to which the sequence x_1^n is divided by the algorithm LZ77. By the construction, f_1 is a prefix coding.

The difference between the algorithm described above and LZ78 [19] is that in the latter, at each step we choose the longest prefix of $x_{l_i}^n$ coinciding with some subword σ_j, $j < i$, and add a letter to it, i. e., $\sigma_{i+1} = \sigma_j a_{p_i}$. The code of a subword σ_{i+1} is defined as the pair (j, p_i). For example, the sequence $a_2a_1a_2a_1a_1a_2a_1a_2a_1$ is divided to subwords $a_2, a_1, a_2a_1, a_1a_2, a_1a_2a_1$ and is coded by the sequence of pairs $(0,2),(0,1),(1,1),(2,2),(4,1)$. The LZ78 coding f_2 is defined as the sequence of pairs (j,s), where the first number of the ith pair is written by $\lfloor \log i \rfloor + 1$ binary bits, and the second one by $\lfloor \log|A| \rfloor + 1$ binary bits. Then

$$|f_2(x_1^n)| = \sum_{i=1}^{m} (\log i + \log|A| + 2) \leqslant m(\log m + \log|A| + 2), \tag{9}$$

where m is the number of subwords σ_i to which x_1^n is divided by LZ78. By the construction, f_2 is a prefix coding.

The modification of the Lempel–Ziv algorithm offered here (and from now on denoted by LZP) is based on both methods LZ77 and LZ78. When choosing the next subword, we always use LZ78 or LZ77; the latter is chosen only if $n_i < r_i - l_i$. The first or the second method is pointed by 0 or 1, respectively; each time we select the method that extracts the longer subword. The subwords are coded by the same way as in LZ78 and LZ77; the only difference is that in the second case, $\lceil \log(r_i - l_i) \rceil$ bits are used to code n_i. For example, the word $a_1a_2a_1a_1a_1a_1a_1a_1a_2$ will be divided into the words $a_1, a_2, a_1a_1, a_1a_1a_1a_1a_2$ and coded by three triples and a quadruple $(0,0,1),(0,0,2),(0,1,1),(1,4,2,2)$. Clearly, the length of the code $|f_3(x_1^n)|$ in the LZP algorithm is bounded as follows:

$$|f_3(x_1^n)| \leqslant m_1 \log m + 2\sum_{\sigma_i \in B_2} (\log|\sigma_i| + \log(1 + \log|\sigma_i|)) + m(\log|A| + 3), \tag{10}$$

where m_1 is the number of subwords obtained by the first method, B_2 is the set of all subwords obtained by the second method, and $m = m_1 + |B_2|$ is the number of all subwords into which the word x_1^n is divided. The coding f_3 is a prefix one by the construction, as well as f_1 and f_2.

All subwords of x_1^n obtained by the algorithms LZ77, LZ78, and LZP are distinct, possibly except for the last subword, which can coincide with one of the previous words. In what follows, to simplify the calculations, we suppose that all the subwords, including the last, are distinct.

3. Main results

The following lemma will be used to estimate the coding redundancy of the algorithm LZP for sequences with asymptotically zero entropy.

Lemma 1. *If $x_1^n \in A^n$ and $x_1^n = \sigma_1 \dots \sigma_m$ for the words σ_i, $1 \leqslant i \leqslant m$, selected by the algorithm* LZP, *then*

$$nH_k(x_1^n) \geqslant \sum_{|\sigma_i| > |A|^k} \max(\log(|\sigma_i|/|A|^k), 1).$$

Proof. Let us denote $\sigma_i' = z_i\sigma_i$, where z_i is the subword consisting of k letters standing directly before σ_i in x_1^n (for several initial σ_i, the subwords z_i may consist of fewer letters). Suppose that $y \in A^k$ and $\sigma_i'(y)$ contains different letters. Then it follows from (3) that

$$|\sigma_i'(y)|H(\sigma_i'(y)) \geqslant \log|\sigma_i'(y)|. \tag{11}$$

Since $\sum_{y \in A^k} |\sigma_i'(y)| = |\sigma_i|$ and $|\sigma_i| > |A|^k$, there exists a $y \in A^k$ such that

$$|\sigma_i'(y)| \geqslant |\sigma_i|/|A|^k, \quad |\sigma_i'(y)| \geqslant 2. \tag{12}$$

If $\sigma_i'(y)$ contains the last letter of σ_i, then, according to the LZP algorithm, $\sigma_i'(y)$ contains at least two different letters, and inequality (11) holds. Let $\sigma_i'(y)$ do not contain the last letter of σ_i and be a power of some one letter $a_{i(y)}$. Let $y = a_{i_1}a_{i_3} \dots a_{i_k}$. Consider the word $y_1 = a_{i_2}a_{i_3} \dots a_{i_k}a_{i(y)}$. Then from the definition of y_1 it follows that $|\sigma_i'(y_1)| \geqslant |\sigma_i'(y)| \geqslant \max(|\sigma_i|/|A|^k, 2)$. If y_1 is not a power of one letter, then (11) holds. Otherwise, let us define the letters y_2, y_3, y_4, and so on similarly to y_1; that is, $y_2 = a_{i_3} \dots a_{i_k}a_{i(y)}a_{i(y_1)}$, etc. Since $y_j \in A^k$, the sequence $y_1, y_2, y_3, \dots$ is either finite or periodic. If it is periodic, then all k-element blocks of the subword σ_i are elements of $y_1, y_2, y_3, \dots$, contradicting the LZP algorithm of choosing σ_i. Thus, if $|\sigma_i| > |A|^k$, then there exists a word $y \in A^k$ for which (11) and (12) hold. □

Since the function $\log x$ is convex, it follows from the Jensen inequality and the definition of entropy (3) that

$$|x_1^n(y)|H(x_1^n(y)) \geqslant \sum_{i=1}^{m} |\sigma_i'(y)|H(\sigma_i'(y)).$$

Then (5), (11), (12), and the last inequality imply the statement of Lemma 1. □

The redundancy estimate for the Lempel–Ziv coding is based on the following statement:

Lemma 2. *Let $x_1^n \in A^n$ and $x_1^n = \sigma_1\sigma_2\dots\sigma_m$, where $\sigma_i \neq \sigma_j$ for $i \neq j$. Then for each integer $k \geqslant 0$ the inequality*

$$nH_k(x_1^n) \geqslant m\log m - \sum_{i=1}^{m}\log|\sigma_i| - 2\sum_{i=1}^{m}\log(1+\log|\sigma_i|) - Cm$$

holds, where the constant $C > 0$ depends only on k and $|A|$.

Proof. Let $a_0 \neq a_i$, $1 \leqslant i \leqslant |A|$. Denote $\hat{A}=A\cup a_0$ and $z_1^k=a_0a_0\dots a_0$, and also $\hat{\sigma}_i=z_1^k\sigma_i$ and $\hat{x}_1^n = \hat{\sigma}_1\dots\hat{\sigma}_m$. Let S_m be the set of permutations of length m. Let $\tau \in S_m$. Denote $\tau(\hat{x}_1^n) = \hat{\sigma}_{\tau(1)}\dots\hat{\sigma}_{\tau(m)}$. Then $\tau(\hat{x}_1^n) \neq \tau'(\hat{x}_1^n)$ for $\tau \neq \tau'$ because $a_0 \in \hat{A}\setminus A$ and the words $\tau(\hat{x}_1^n)$, $\tau'(\hat{x}_1^n)$ are uniquely divided into subwords σ_i which are all distinct by assumption.

Consider $y \in \hat{A}^k$ and the word $\hat{\sigma}_1(y)\hat{\sigma}_2(y)\dots\hat{\sigma}_m(y)$. If $\tau \in S_m$, then it follows that $\hat{\sigma}_{\tau(1)}(y)\hat{\sigma}_{\tau(2)}(y)\dots\hat{\sigma}_{\tau(m)}(y)$ is a permutation $\delta_\tau(y)$ of letters of the word $\hat{\sigma}_1(y)\hat{\sigma}_2(y)\dots\hat{\sigma}_m(y)$. Let $\Delta(y)$ be the set of all such permutations $\delta_\tau(y)$ for $\tau \in S_m$. If $y \in A^k$, then $\hat{\sigma}_1(y)\hat{\sigma}_2(y)\dots\hat{\sigma}_m(y)=\sigma_1(y)\sigma_2(y)\dots\sigma_m(y)$ and $|\sigma_1(y)\sigma_2(y)\dots\sigma_m(y)| \leqslant |x_1^n(y)|$. Moreover, the number of occurrences of a letter $a_i \in A$, $1 \leqslant i \leqslant |A|$, to $\sigma_1(y)\sigma_2(y)\dots\sigma_m(y)$ is not greater than the number of its occurrences to $x_1^n(y)$. Thus,

$$|\Delta(y)| \leqslant \frac{n(y)!}{r_1(y)!r_2(y)!\dots r_{|A|}(y)!}, \tag{13}$$

where $n(y) = |x_1^n(y)|$ and $r_i(y)$ is the number of occurrences of $a_i \in A$ to $x_1^n(y)$. If $y \in \hat{A}^k \setminus A^k$, then $|\hat{\sigma}_1(y)\hat{\sigma}_2(y)\dots\hat{\sigma}_m(y)| \leqslant m$ and

$$|\Delta(y)| \leqslant |\hat{A}|^m. \tag{14}$$

Each word $\tau(\hat{x}_1^n)$ from S_m can be uniquely assigned by fixing the number of letters between consecutive subwords z_1^k in the word $\tau(\hat{x}_1^n)$: $|\hat{\sigma}_{\tau(1)}|, |\hat{\sigma}_{\tau(2)}|, \dots, |\hat{\sigma}_{\tau(m-1)}|$ and fixing the permutations $\delta_\tau(y) \in \Delta(y)$ for all $y \in \hat{A}^k$. This, (6), and the equality $|S_m| = m!$ imply

$$\log m! \leqslant \sum_{y\in\hat{A}^k}\lceil\log|\Delta(y)|\rceil + \sum_{i=1}^{m-1}\lfloor\log|\sigma_i|\rfloor + 2\sum_{i=1}^{m-1}\lfloor\log(1+\log|\sigma_i|)\rfloor + m. \tag{15}$$

It follows from the definition of $\hat{\sigma}_i$ that the number of different words $y \in \hat{A}^k \setminus A^k$ contained in $\hat{\sigma}_i$ is not greater than $k|A|$. Using (4), (13), (14), we get

$$\sum_{y\in\hat{A}^k}\lceil\log|\Delta(y)|\rceil \leqslant \sum_{y\in A^k}\left\lceil\log\frac{n(y)!}{r_1(y)!r_2(y)!\dots r_{|A|}(y)!}\right\rceil + k|A|\lceil\log(|\hat{A}|^m)\rceil$$

$$\leqslant \sum_{y\in A^k} n(y)H(x_1^n(y)) + km|A|\log|\hat{A}| + (|A|+1)^k.$$

The statement of Lemma 2 now follows from (5), (15), the inequality $\log m! \geqslant m \log m - m/\ln 2$ and the last inequality. □

In the next lemma, we obtain a lower bound for the number of distinct subwords to which the word can be divided.

Lemma 3. *Let* $\sigma_1, \sigma_2, \ldots, \sigma_m \in A^*$, *where* $\sigma_i \neq \sigma_j$ *for* $i \neq j$, *then*

$$\frac{1}{m}\sum_{i=1}^{m} |\sigma_i| \geqslant \frac{\log m}{2\log|A|} - 1.$$

Proof. Since the number of the words σ_i of length k is not greater than $|A|^k$, we have

$$\sum_{k=1}^{\lfloor \log_{|A|} m \rfloor - 2} \sum_{|\sigma_i|=k} 1 \leqslant \sum_{k=1}^{\lfloor \log_{|A|} m \rfloor - 2} |A|^k \leqslant \frac{m}{|A|}.$$

So,

$$\sum_{k=\lfloor \log_{|A|} m \rfloor - 1}^{\infty} \sum_{|\sigma_i|=k} 1 \geqslant m - \frac{m}{|A|} \geqslant \frac{m}{2}$$

and

$$\frac{1}{m}\sum_{i=1}^{m} |\sigma_i| \geqslant \frac{1}{m}\frac{m}{2}(\lfloor \log_{|A|} m \rfloor - 1) \geqslant \frac{\log m}{2\log|A|} - 1.$$

The lemma is proved. □

Now let us estimate the redundancy of coding the sequence having asymptotically positive entropy.

Theorem 1. *Consider* $A = \{a_1, \ldots, a_{|A|}\}$, $x \in A^\infty$, *and an integer* $k \geqslant 0$ *such that* $\lim_{n\to\infty} H_k(x_1^n) > 0$. *Then*

$$R_k(f, x_1^n) \leqslant \frac{CH_k(x_1^n)\log\log n}{\log n}(1 + o(1)),$$

where $C = 1$ *for the codings built according to the rules* LZ78 *and* LZP, *and* $C = 3$ *for the coding built according to* LZ77.

Proof. Let the algorithm LZP divide a word $x_1^n \in A^n$ to subwords $\sigma_1, \sigma_2, \ldots, \sigma_m$. Since $\sum_{i=1}^{m} |\sigma_i| = n$, and $\log x$ is a convex function, it follows from the Jensen inequality that

$$\sum_{i=1}^{m} \log|\sigma_i| \leqslant m \log\left(\sum_{i=1}^{m} |\sigma_i|/m\right) = m \log\frac{n}{m}. \quad (16)$$

Analogously, since $\log\log x$ is a convex function, we have

$$\sum_{i=1}^{m}\log(1+\log|\sigma_i|)\leqslant m\left(1+\log\log\frac{n}{m}\right). \tag{17}$$

Now (10), (16) and (17) for the coding f built by the rule LZP imply

$$|f(x_1^n)|\leqslant m_1\log m+2m_2\log\frac{n}{m_2}+2m\log\log\frac{n}{m}+m(\log|A|+5), \tag{18}$$

where m_1 and m_2 are the numbers of subwords selected from x_1^n by the first and the second rule, respectively.

Since $-t\log(t/c)\leqslant c$ for $c>0$ and $t>0$, we have $2m_2\log n/m_2\leqslant m_2\log m+2(n/\sqrt{m})$. Since all σ_i are mutually distinct by the construction, it follows from Lemma 2, (7),(16)–(18) and the last inequality that

$$R_k(f,x_1^n)\leqslant\frac{m}{n}\log\frac{n}{m}+4\frac{m}{n}\log\log\frac{n}{m}+C'\frac{m}{n}+\frac{2}{\sqrt{m}}, \tag{19}$$

where $C'>0$ is a constant.

Lemma 3 implies that $m/n\to 0$ as $m\to\infty$. Using (19), we obtain that $\limsup_{n\to\infty} R_k(f,x_1^n)\leqslant 0$. Without loss of generality we may assume that $\limsup_{n\to\infty} R_k(f,x_1^n)=\lim_{n\to\infty} R_k(f,x_1^n)$ (if it is necessary, we pass to an appropriate subsequence). If $\lim_{n\to\infty} R_k(f,x_1^n)<0$, then the theorem obviously holds. Otherwise, it follows from $\lim_{n\to\infty} H_k(x_1^n)>0$ and $\lim_{n\to\infty} R_k(f,x_1^n)=0$ that $H_k(x_1^n)\sim(1/n)|f(x_1^n)|\sim(m/n)\log m$ and $H_k(x_1^n)/\log m\sim m/n$, $\log m/H_k(x_1^n)\sim n/m$ for $n\to\infty$. These equivalences and (19) imply the theorem for the coding built by the rule LZP. For the codings constructed by LZ77 and LZ78, the theorem can be proved analogously with the use of Lemma 2, (8), and (9). Theorem 1 is proved. □

Let us consider an example of the sequence x such that

$$|f(x_1^n)|/nH_1(x_1^n)\to\infty \tag{20}$$

as $n\to\infty$, where the coding f is built by LZ78.

Let x be the sequence on 2 letters defined as follows: $x_i=a_1$ if $i=2^k$ for some integer k, and $x_i=a_2$ otherwise. Then it follows from (3) and (5) that $H_1(x_1^n)=(\log^2 n/n)(1+o(1))$. The algorithm LZ78 divides this sequence to at least $\sqrt{n}$ subwords, since the lengths of subwords cannot grow faster than the members of some arithmetic progression. Then it follows from (9) that

$$|f(x_1^n)|=\frac{\sqrt{n}}{2}\log n(1+o(1))$$

for the coding f, i. e., that

$$|f(x_1^n)|/nH_1(x_1^n)=\frac{\sqrt{n}}{2\log n}(1+o(1))\to\infty$$

as $n\to\infty$.

The examples of sequences whose code satisfies (20) also exist for the rule LZ77. But as the following theorem shows, for the code built by LZP the limit from (20) is always finite.

Theorem 2. *Let* $A = \{a_1, \dots, a_{|A|}\}$, $x \in A^\infty$, $\lim_{n\to\infty} H_k(x_1^n) = 0$, *and let the sequence* x_i^∞ *be non-periodic for each integer* $i > 0$. *Then*

$$\frac{1}{n}|f(x_1^n)| = O(H_k(x_1^n)),$$

where the coding f *is built according to the* LZP *rule.*

Proof. Let the LZP algorithm divide the word $x_1^n \in A^n$ to subwords $\sigma_1, \sigma_2, \dots, \sigma_m$. Since for each integer $i > 0$ the sequence x_i^∞ is non-periodic, the procedure of selecting the next word by the LZP algorithm is always finite. Let the coding $f : A^* \to E^*$ be built by the LZP rule. Due to Lemma 2 and (10), for each integer $k \geqslant 0$ we have

$$|f(x_1^n)| - nH_k(f, x_1^n) \leqslant 6\sum_{i=1}^{m} \log|\sigma_i| + Cm, \tag{21}$$

where $C > 0$ depends only on k and $|A|$. Due to Lemma 1, we have

$$nH_k(f, x_1^n) \geqslant \sum_{i=1}^{m} \log|\sigma_i| - mk\log|A| - |A|^{2k}\log|A|$$

and

$$m \leqslant nH_k(f, x_1^n) + |A|^k \log|A|.$$

Now Theorem 2 follows from (21) and the two last inequalities. □

References

[1] R.P. Brent, A linear algorithm for data compression, Austral. Comput. J. 19 (2) (1987) 64–68.

[2] P. Elias, Universal codeword sets and representations of integers, IEEE Trans. Inform. Theory 21 (2) (1975) 194–203.

[3] V.D. Goppa, Codes and Information, Uspehi Mat. Nauk 39 (1) (1984) 77–120 (in Russian).

[4] A.V. Kadach, Effective algorithms of non-distorting text data compression, Candidate Thesis, Novosibirsk, Ershov Institute of Informatics Systems, 1997 (in Russian).

[5] J. Kieffer, W. Finamore, P. Nunes, A class of noiseless data compression algorithms based on Lempel–Ziv parsing trees, in: Proceedings of the IEEE International Symposium on Information Theory, Piscataway, NJ, 1994, p. 6.

[6] R.E. Krichevsky, Universal Compression and Retrieval, Kluwer Academic Publishers, Dordrecht, 1994.

[7] V.I. Levenstein, On redundancy and deceleration of separably coding positive integers, Problemy Kibernetiki 20 (1968) 173–179 (in Russian).

[8] G. Lounchard, W. Szpankowski, On the average redundancy rate of the Lempel–Ziv code, IEEE Trans. Inform. Theory 43 (1) (1997) 1–7.

[9] V.S. Miller, M.N. Wegman, Variations on a theme by Ziv and Lempel, in: Combinatorial Algorithms on Words, Springer, Berlin, 1985, pp. 131–140.

[10] E. Plotnick, M.J. Weinberger, J. Ziv, Upper bounds on the probability of sequences emitted by finite-state source and on the redundancy of the Lempel–Ziv algorithm, IEEE Trans. Inform. Theory 38 (1) (1992) 66–72.

[11] S.A. Savari, Redundancy of the Lempel–Ziv incremental parsing rule, IEEE Trans. Inform. Theory 43 (1) (1997) 8–16.
[12] S.A. Savari, Redundancy of the Lempel–Ziv string matching code, IEEE Trans. Inform. Theory 44 (2) (1998) 787–792.
[13] J.A. Storer, T.G. Szymansky, Data compression via textual substitution, J. Assoc. Comput. Mach. 25 (4) (1982) 928–951.
[14] T.A. Welch, A technique for high-performance data compression, IEEE Comput. 17 (6) (1984) 8–19.
[15] A.J. Wyner, The redundancy and distribution of phrase lengths of the fixed-database Lempel–Ziv algorithm, IEEE Trans. Inform. Theory 43 (5) (1997) 1452–1464.
[16] A.D. Wyner, A.J. Wyner, Improved redundancy of a version of the Lempel–Ziv algorithm, IEEE Trans. Inform. Theory 41 (3) (1995) 723–731.
[17] A.D. Wyner, J. Ziv, Fixed data base version of the Lempel–Ziv algorithm, IEEE Trans. Inform. Theory 37 (3) (1991) 723–731.
[18] J. Ziv, A. Lempel, A universal algorithm for sequential data compression, IEEE Trans. Inform. Theory 23 (3) (1977) 337–343.
[19] J. Ziv, A. Lempel, Compression of individual sequences via variable-length coding, IEEE Trans. Inform. Theory. 24 (5) (1978) 530–536.