The Imaginary Sliding Window As a New Data Structure for Adaptive Algorithms

Boris Ryabko

Siberian State University
of Telecommunication and Computer Science.

Abstract.[1] The scheme of the sliding window is known in Information Theory, Computer Science, the problem of predicting and in stastistics. Let a source with unknown statistics generate some word $\ldots x_{-1}x_0x_1x_2\ldots$ in some alphabet A. For every moment $t, t = \ldots -1, 0, 1, \ldots$, one stores the word ("window") $x_{t-w}x_{t-w+1}\ldots x_{t-1}$ where w,$w \geq 1$, is called "window length". In the theory of universal coding, the code of the x_t depends on source ststistics estimated by the window, in the problem of predicting, each letter x_t is predicted using information of the window, etc. After that the letter x_t is included in the window on the right, while x_{t-w} is removed from the window. It is the sliding window scheme. This scheme has two merits: it allows one i) to estimate the source statistics quite precisely and ii) to adapt the code in case of a change in the source' statistics. However this scheme has a defect, namely, the necessity to store the window (i.e. the word $x_{t-w}\ldots x_{t-1}$) which needs a large memory size for large w. A new scheme named "the Imaginary Sliding Window (ISW)" is constructed. The gist of this scheme is that not the last element x_{t-w} but rather a random one is removed from the window. This allows one to retain both merits of the sliding window as well as the possibility of not storing the window and thus significantly decreasing the memory size.
Index Terms: data structure, storage and search of information, prediction, randomization, data compression.

1 Introduction

There are many situations when people deal with information sources with unknown or changing ststistics. Among them we mention data compression [2], the problem of predicting [14] and the similar problem of prefetching [15], the problem of adaptive search [9] as well as a statistical estimation of parameters of the information sources. There are many interesting ideas and algorithms for solving these problems. For example, for encoding of information sources with unknown or changing statistics different methods are used to adapt a code to statistics of a source. Many of such methods are based on context - tree weighting procedure [10], on Lempel - Ziv codes [16], see also the review in [2] , a bookstack scheme [11][2], on a scheme of sliding window and some others.

[1]Supported by Russian Foundation of Fundamental Research under Grant 99-01-00586

Some of these methods are used not only for information sources encoding, but also for solving problems connected with storage and search of the information, see [1], [9], [15] as well as with statistical estimation of parameters of changing random processes.

The scheme of sliding window is quite popular and may be used jointly with the adaptive Huffman code [7], the arithmetic code [2], the interval code [4], the fast code [13], as well as for predicting [14] and prefetching [15] and other algorithms.

Let us define the sliding window scheme (SW). A source generates the word $\ldots x_{-1}x_0x_1x_2\ldots$ in a finite alphabet A. There is a computer which stores the window $x_{t-w}x_{t-w+1}\ldots x_{t-1}$ of the moment t, where $w (w \geq 1)$ is the length of the window. The computer uses the window in order to estimate the source statistics. After that the computer moves the window as follows: the letter x_t is included in the window on the right, while the letter x_{t-w} is removed from the window. If the SW is used for data compression there exist two computers (an encoder and a decoder). The decoder conducts the same operation with the window and this allows to decode a message definitely. Naturally, the greater w, the more precise the estimate of statistics of the source.

Let us consider as an example the problem of encoding the Bernoulli source generating letters from some alphabet $A = \{a_1, a_2, \ldots, a_m\}$. In this case, the redundancy per letter for the best universal code is not less than $(m-1)/2w + O(1/w)$ when w is the length of the window (see [8]). (In case of predicting and prefetching the redundancy is equal to the precision of the prediction). Hence, to achieve smaller redundancy or precision, the length of the window has to be much greater than the number of letters in the alphabet A. Obviously, to keep the window, the encoder and the decoder need $w \log m$ bits of memory.

By $\nu^t(a)$ we denote the frequency of occurence of the letter a in the window $x_{t-w}\ldots x_{t-1}$ for every $a \in A$. It is known that a vector $\{\nu^t(a), a \in A\}$ is a sufficient statistic for the Bernoulli source (see [6] for example). Informally, this means that this vector implies that all the information is contained in $x_{t-w}\ldots x_{t-1}$. However to keep frequencies only $m \log w$ bits are needed (which is exponentially less then $w \log m$). In the sliding window scheme after the encoding of the recurrent letter x_t the relevant frequency increases by 1 $(\nu^{t+1}(x_t) = \nu(x_t) + 1)$ while the frequency of the letter x_{t-w}, being removed from the window, decreases by 1.

Informally, in the ISW scheme, we propose only to keep the vector of frequencies $(\bar\nu^t(a_1), \bar\nu^t(a_2), \ldots, \bar\nu^t(a_m))$, $\sum_{i=1}^m \bar\nu^t(a_i) = w$. After the coding of the recurrent letter x_t we increase its occurrence by 1 (as before, $\bar\nu^{t+1}(x_t) = \bar\nu^t(x_t) + 1$) and then we decrease the occurrence of a randomly selected letter by 1, where the probability of decreasing the occurrence of the letter $a \in A$ is equal to $\bar\nu^t(a)/w$.

Thus, in the Imaginary Sliding Window scheme only the vector of integers $(\bar\nu^t(a_1), \ldots, \bar\nu^t(a_m))$ is kept. After the encoding of x_t the number $\bar\nu^t(x_t)$ increases by 1 and one randomly selected number decreases by 1.

It turns out that this scheme possesses properties which are similar to the usual sliding window : first, the distribution of the vector $(\bar\nu^t(a_1), \ldots, \bar\nu^t(a_m))$ is similar to the one for the sliding window and, second, under the changing of the source statistics, the adaptation of the vector $(\bar\nu^t(a_1), \ldots, \bar\nu^t(a_m))$ occurs.

---

[2]The bookstack scheme was proposed in authors' paper [11] and then rediscovered in 1986-1987 in [3] and [4] (see also [12]). In Western literature this scheme is usually nemed "move-to-front scheme" as proposed in [3].

The construction of the ISW scheme is based on the use of random numbers, or, more precisely, on a sequence of random equi - probable independent binary digits.

There are two usual ways to obtain random digits: the first, using a table of random digits, and, the second, using a generator of random digits. Both of them may be used for predicting and statistical estimation. However in case of data compression the sequence of digits has to be the same in the encoder and the decoder. It is possible to use the third way to obtain random digits. In this case the random digits are not ideal, but they may be obtained free "of charge". The point is that the encoded, "compressed" sequence is "nearly" random. Moreover, the less the code redundancy, the nearer encoded message is to a sequence of random bits generated by tossing a symmetric coin. We recommend to use as random digits that part of the message being generated by a source, which is already encoded. It is important that this sequence is known by the encoder and the decoder, and while coding, it is sufficient to keep only a small current piece of it.

We turn our attention to the remainder of the paper. In Section 2 the asymptotic equivalence of the scheme of the imaginary sliding window with the usual scheme of the sliding window is proved and the extention to the case of Markovian sources is given. In Section 3 a fast method to use random digits which are necessary for the Imaginary Sliding Windows scheme is proposed. Using this method allows the processes of encoding and decoding to procede without delay.

2 The Scheme of the Imaginary Sliding Window

Let us give some necessary definitions. Denote the set of words of the length k in the alphabet A as A^k ,$(k \geq 0)$. Let Ω_∞ be the set of all ergodic and stationary sources generating letters from A. For $\omega \in \Omega_\infty$, $a \in A$, $k \geq 0$, $u \in A^k$ denote by $P_\omega(a/u)$ the probability that letter a is generated next by the source ω in the case when the word $u \in A^k$ is generated by the same source. According to the definition, μ will be the memory of the source in the case if for all letters a, $u_1, u_2, \ldots, u_k \in A$, $k \geq \mu$ the equality

$$P_\omega(a/u_1 \ldots u_k) = P_\omega(a/u_1 \ldots u_\mu)$$

is valid (when $\mu = 0$ a source is said to be a Bernoulli source). Denote as Ω_μ, $\mu \geq 0$, a set of all $\omega \in \Omega_\infty$ with the memory μ.

Let us describe the scheme of the ISW for the case of a Bernoulli source. Let $x_1 x_2 \ldots x_t \ldots$ be the sequence being generated by some Bernoulli source $\omega \in \Omega_0$. Let w, $w \geq 1$, be the window length and let for any integer t a word $x_{t-w} x_{t-w+1} \ldots x_{t-1}$ be the window at the moment t.

Denote as ν_j^t the number of occurences of the letter $a_j \in A$ in the window $x_{t-w} \ldots x_{t-1}$. It is easy to see that $(\nu_1^t, \ldots, \nu_m^t)$ is a random vector governed by the multinomial distribution :

$$P\{\nu_1^t = n_1, \nu_2^t = n_2, \ldots, \nu_m^t = n_m\} = \left(\begin{array}{c} w \\ n_1 n_2 \ldots n_m \end{array} \right) \prod_{i=1}^{m} P(a_i)^{n_i} \quad (1)$$

In the construction of the ISW only the set of integers (without a window) $\bar{\nu}^t = (\bar{\nu}_1^t, \bar{\nu}_2^t, \ldots, \bar{\nu}_m^t)$ is stored, which changes after encoding of every letter x_t. To describe the rules of changing the vector $\bar{\nu}^t$ let us denote the random value ε^t being the vector $1, 2, \ldots, m$ with the probabilitis $\bar{\nu}_1^t/w, \bar{\nu}_2^t/w, \ldots, \bar{\nu}_m^t/w$, respectively, i.e.

$$P\{\varepsilon^t = i\} = \bar{\nu}_i^t/w, \quad i = 1, 2, \ldots, m \quad (2)$$

After encoding every letter x_t of the message $x_1 x_2 \ldots x_t$, first, ε^t is generated and then the conversion from the vector $\bar{\nu}^t$ to the vector $\bar{\nu}^{t+1}$ is conducted :

$$\bar{\nu}_j^{t+1} = \begin{cases} \bar{\nu}_j^t - 1, & \text{if} \quad j = \varepsilon^t \\ \bar{\nu}_j^t + 1, & \text{if} \quad a_j = x_t \\ \nu_j^t, & \text{if } j \neq \varepsilon^t \text{ and } a_j \neq x_t \text{ or } j = \varepsilon^t \text{ and } x_t = a_j \end{cases} \tag{3}$$

In other words, firstly, one random chosen coordinate of the vector $\bar{\nu}^t$ is decreased by 1. (This operation is analogous to decreasing a counter which corresponds to x_{t-w} , by 1, when the window moves from $x_{t-w} \ldots x_{t-1}$ to $x_{t-w+1} \ldots x_t$ and, instead of removing x_{t-w}, one random chosen letter is "thrown out" from the window). Second, the coordinate of the vector $\bar{\nu}^t$ corresponding to the letter x_t is increased by 1.

The initial distribution $\bar{\nu}^0 = (\bar{\nu}_1^0, \ldots, \bar{\nu}_m^0)$ may be arbitrary chosen. For example, $\bar{\nu}_i^0 = w/m$ may be assumed when $i = 1, \ldots m$ if w/m is integer.

Now, we investigate the propeties of the ISW. First, we demonstrate that the distribution of the vector $\bar{\nu}^t$ asymptotically complies with (3), i.e., it is the same as the distribution of the frequency of the occurence of letters in the scheme of the sliding window.

Theorem 1. Let a Bernoulli source be given which generates letters from the alphabet $A = \{a_1, \ldots, a_m\}$ with probabilities $P(a_1), \ldots, P(a_m)$, and let $n_1 n_2, \ldots, n_m$ be any integer nonnegative numbers such as $\sum n_i = w$, $w \geq 1$. Then for the scheme of the ISW with the vector of frequencies $\bar{\nu} = (\bar{\nu}_1^t, \ldots, \bar{\nu}_m^t)$ the following equality is valid for any initial vector $\bar{\nu}^0 = (\nu_1^0, \ldots, \nu_m^0)$:

$$\lim_{t \to \infty} P\left\{\bar{\nu}_1^t = n_1, \ldots, \bar{\nu}_m^t = n_m\right\} = \binom{w}{n_1, n_2 \ldots, n_m} \prod_{i=1}^{m} P(a_1)^{n_i}$$

Proofs of all theorems are given in the appendix.

Hence, values $(\bar{\nu}_1^t, \bar{\nu}_2^t, \ldots, \bar{\nu}_m^t)$ may replace values of frequency of occurence of letters in an usual sliding window and then the ISW may asymptotically replace the SW.

The rate of convergence of the distribution $(\bar{\nu}_1^t, \ldots, \bar{\nu}_m^t)$ to multinomial distribution (1) come into the question. This is of importance, because this rate effects the rate of adaptation of the ISW to modifications of statistics. (In fact, we may assume the statistics change at the moment $t = 0$) We mention two conclusions characterizing the rate of approximation of frequencies $(\bar{\nu}_1^t, \ldots, \bar{\nu}_m^t)$ to the limit distribution, presupposing that the vector $(\bar{\nu}_1^0, \ldots, \bar{\nu}_m^0)$ is chosen arbitrarily. For simplicity sake, let us define

$$P\left\{\bar{\nu}_1^\infty = n_1,\ \bar{\nu}_2^\infty = n_2, \ldots, \bar{\nu}_m^\infty = n_m\right\} = \binom{w}{n_1, n_2, \ldots, n_m} \prod_{i=1}^{m} P(a_i)^{n_i} \tag{4}$$

(Such a definition is based on Theorem 1).

In Information Theory and Statistics there is well known the Kullback-Leibler Divergence estimating the divergence of the two distributions of probabilities. The next Theorem allows estimating the divergence of the distribution $(\bar{\nu}_1^t, \ldots, \bar{\nu}_m^t)$ to (4).

Theorem 2. Suppose a Bernoulli source generating letters from the finite alphabet $A = \{a_1, \ldots, a_m\}$ is given and we use the scheme of ISW with the "window length" w. Let R^t be Kullback-Leibler Divergence between distributions of probabilities of the vector of frequencies $(\bar{\nu}_1^t, \ldots, \bar{\nu}_m^t)$ and $(\bar{\nu}_1^\infty, \ldots, \bar{\nu}_m^\infty)$, defined by the equation

$$R^t = \sum_{(n_1, \ldots n_m)} P\left\{\bar{\nu}_1^\infty = n_1, \ldots, \bar{\nu}_m^\infty = n_m\right\} \log \frac{P\left\{\bar{\nu}_1^\infty = n_1, \ldots, \bar{\nu}_m^\infty = n_m\right\}}{P\left\{\bar{\nu}_1^t = n_1, \ldots, \bar{\nu}_m^t = n_m\right\}} \tag{5}$$

Then, under any initial distribution of frequencies $(\bar{\nu}_1^0, \ldots, \bar{\nu}_m^0)$ the inequality

$$R^t \le -\log \left(\sum_{k=0}^{w} \binom{w}{k} (-1)^k \left(1 - \frac{k}{w}\right)^t \right) \tag{6}$$

is valid.

The right part in (6) is rather cumbersome. For large t and w the following asymptotic estimate is valid:

Corollary. Let $t \to \infty$ and let

$$\lambda = w \ e^{-t/w} \tag{7}$$

Then $R < \lambda + o(\lambda)$.

It readily follows from the corollary above that R^t becomes small when $t > w \log w$. If, for example, $t = w \log w + bw$ then R^t is close to e^{-b}.

Thus, the ISW "remembers" the initial distribution of probabilities during a period which is approximately equal to $w \log w$. An "usual" SW can "remember" the initial distribution of probabilities till total renewal of its contents, i.e. till $t = w$.

To encode a source as well as to use the schemes of SW (and ISW) in many other applications the estimates of probabilities $P(a_1), \ldots P(a_m)$ are used and the values $\bar{\nu}_i^t / w$, $i = 1, \ldots, m$ (or similar ones) are used as these estimates. The next Theorem allows estimation of the proximity of $\bar{\nu}_i^t / w$ to $P(a_i)$

Theorem 3. Under fulfilment of the hypothesises of the Theorem 2,

$$\mid E(\bar{\nu}_i^t / w) - P(a_i) \mid < e^{-t/w} \quad \text{for } \ i = 1, \ldots, m.$$

It readily follows from this that the average value of estimates of probabilities of the letters $a \in A$ obtained by using ISW, quite rapidly approximate to the proper value of $P(a)$ under increasing t. It is important for application of ISW because t may be interpreted as the time duration after modification of statistics (at the moment $t = 0$).

Now, we shall apply the scheme of ISW to the case of Markovian sources. Let $\mu \ge 1$ and it is known that $\omega \in \Omega_\mu$. The construction of ISW described above may be applied to this case in such a way as while encoding and decoding, we store $\mid A \mid^\mu$ imaginary windows and each of them corresponds to one word from A^μ. Futhermore, in the memory of the encoder and the decoder one "real" window is kept, consisting of μ letters, and the last μ letters encoded are stored in this window. For example, let a source generate the message $x_1 x_2 \ldots x_t \ldots$ Then, before encoding x_t there are letters $x_{t-\mu} \ldots x_{t-1}$ stored in the "real" window. This word belongs to A^μ, hence, the ISW correspondsing to it exists and the letter x_t is encoded in accordance with the information stored in this window. After encoding of x_t, the same mapping are made with the ISW which corresponds to $x_{t-\mu} \ldots x_{t-1}$, as in the Bernoulli case described above. (One randomly chosen frequency decreases at 1, and a frequency corresponding to x_t increases at 1).

Let us consider an example which explains the described construction. Let $A = \{0, 1\}$, $\mu = 2$ and let 001011 be the sequence being encoded. The encoder and the decoder keep in their memory $2^2 = 4$ imaginary windows, and each of them consists of two nonnegative numbers which, in sum, are equal to the "window length" w (w is any positive number). The first number corresponds to the frequency of occurence of the letter 0 in the window,and the second number corresponds to the frequency of occurence of the letter 1. The letter x_3 which follows after 00 is coded on the basis of the window corresponding to the word 00, the letter x_4 is coded on the basis of the contents of the window 01, etc.

Thus, while encoding a source of memory μ we use the method which is well known in Information Theory : represent the Markovian source as a population of Bernoulli sources. Due to this, every letter generated by a source,is encoded and decoded according to the information which is stored in the window corresponding to the relevant Bernoulli source.

3 Fast Algorithm for Transformation of the ISW

After the coding of every letter of a message transformations of frequences of ISW are conducted: one frequency increases by 1 and another, randomly chosen, decreases by 1. In this section a simple and fast algorithm of realization of random choice is considered.

Let any generator of random bits generate the sequence $z = z_1 z_2 \dots z_k$ which consists of symbols from the alphabet $\{0, 1\}$. We do not estimate the complexity of generating these symbols, and consider only the method of transformation of the random bits to meanings of the random values ε^t (see (4)) which are used for random choice of the frequency being decreased by 1.

Let us give some definitions to start describing an algorithm. For simplicity sake, we shall suppose that the window length w and the number of letters of the alphabet m may be represented as

$$w = 2^u, \quad m = 2^\mu \tag{8}$$

when u and μ are integers. Let $\nu^t = (\nu_1^t, \dots, \nu_m^t)$ be an integer-valued vector characterizing the imaginary window.For generating a meaning of a random value ε^t first, u random bits $z_1 \dots z_u$ are produced, and let

$$z = \sum_{j=1}^{u} z_j 2^{u-j}$$

That shows, along with (8), that z, with the same probability, may be equal to any value from the set $\{0, 1, \dots, w-1\}$, i.e.

$$P\{z = i\} = \begin{cases} 1/w, & \text{if } 0 \le i \le w-1 \\ 0, & \text{for other } i \end{cases} \tag{9}$$

Let us define

$$Q_1 = 0, \quad Q_j = \sum_{k=1}^{j-1} \nu_k^t, \quad j = 2, \dots, m+1 \tag{10}$$

Let us consider the random value ε^t with the meanings j, $1 \le j \le m$ if two inequalities hold:

$$Q_j \le z < Q_{j+1} \tag{11}$$

From this definition follows:

$$P\{\varepsilon^t = j\} = P\{Q_j \le z < Q_{j+1}\} = (Q_{j+1} - Q_j)/w = \nu_j/w$$

(Here the second and the third equalities follow from (9) and (10)). Hence, we obtain

$$P\{\varepsilon^t = j\} = \nu_j^t/w$$

which is the same as the definition (2). From this, it follows that the given method of generating the random value ε^t is quite correct, however it is rather complex. The point is that after encoding of the recurrent letter x_t from the message two frequencies must be changed (one has to be increased by 1, and one has to be decreased by 1). After that, in turn, the values $\{Q_j\}$ must be calculated. In the case of a large source alphabet, calculation of the value Q_j (according to (10)) and searching j (according to (11)) may take too much time. More exactly, $O(m \ \log w)$ operations over one-bit words when $m \to \infty$ are needed.

In conclusion of this section the description of the algorithm which allows carring out all operations with ISW during the period $O(\log m \ \log w)$ for large m and w,is given.This algorithm is close to the fast letter-by-letter code from the author's paper [13].

For description of the method let us define:

$$\Sigma^t_{1,j} = \nu^t_j, \ \ l = 1, 2, \ldots, m; \ \ \Sigma^t_{k,j} = \Sigma^t_{k-1,2j-1} + \Sigma^t_{k-1,2j}, \ \ k = 2, \ldots, \mu; \ \ j = 1, \ldots, m/2^k \tag{12}$$

These values are stored in the memory of the encoder and the decoder. When generating the meaning of the random value ε^t according to z, instead of (11) we use the following algorithm: first, let us check the inequality

$$z \le \Sigma^t_{\mu,1} \tag{13}$$

If it is valid, it means that $1 \le \varepsilon^t \le m/2$, otherwise $m/2 + 1 \le \varepsilon^t \le m$. Then, if the inequality (13) holds, we check the inequality

$$z \le \Sigma^t_{\mu-1,1} \tag{14}$$

Otherwise we calculate $z = z - \Sigma^t_{\mu,1}$ and check the following condition:

$$z \le \Sigma^t_{\mu-1,3} \tag{15}$$

If (14) holds we evaluate whether the inequalities $1 \le \varepsilon^t \le m/4$ or $(m/4+1) \le \varepsilon^t \le m/2$ hold. If (15) holds, we obtain $(m/2) + 1 \le 3m/4$ or $(3m/4 + 1) \le \varepsilon^t \le m$. Continuing in that way, after $\log m = \mu$ steps we shall evaluate ε^t. Besides, at every step, it is necessary to make one comparison and, possibly, one subtraction of numbers each of which has the form of a word of the length $u = \log w$ bits. Thus, the general number of operations over singlebit words is equal to $O(\log m \ \log w)$.

Now, let us describe the "fast" method of conversion from $\left\{\Sigma^t_{i,j}\right\}$ to $\left\{\Sigma^{t+1}_{i,j}\right\}$. Let under conversion from t to $t+1$ any j-coordinate of the vector ν^t increases by 1 and k-coordinate decreases ($j, k \in \{1, 2, \ldots, m\}$, see (5)). Then we have to increase and decrease by 1 one value from the sets

$$\left\{\Sigma^t_{2,i}, \ \ i = \lambda, \ldots, m/2\right\}, \ \left\{\Sigma^t_{3,i}, \ \ i = 1, m/4\right\}, \ \ \ldots, \left\{\Sigma^t_{\mu,i}, \ \ i = 1, 2\right\}$$

i.e. make $\mu = \log m$ operations of addition of 1 and $\mu = \log m$ operations of subtraction of 1. Each operation of addition and subtraction is made over the numbers of length $u = \log w$, so the general number of operations over singlebit words is equal to $O(\log m \ \log w)$. Thus, when using the fast method proposed the number of operations after encoding of a recurrent letter under transformations of ISW, is equal to $O(\log m \ \log w)$.

Appendix

Proof of Theorem 1.

Denote by S a set of vectors of the form of $S = (S_1, \dots, S_m)$ such that all S_j are positive integers, and $\Sigma_{i=1}^m S_j = w$. Let us consider a Markov chain M, states of which coincide with elements of S and a matrix of probabilities of conversion is defined by the equality

$$P_{\sigma,\delta} = \begin{cases} P(a_i)\sigma_j/w \quad \text{if } \sigma_1 = \delta_1, \dots, \delta_i = \sigma_i + 1, \delta_j = \sigma_j - 1 \\ \Sigma_{k=1}^m P(a_k)\delta_k/w \quad \text{if } \delta_1 = \sigma_1, \ \ \delta_2 = \sigma_2, \dots, \delta_m = \sigma_m \\ 0 \quad \text{for another } \sigma, \delta \end{cases} \tag{16}$$

This Markov chain simulates the behaviour of ISW.

Using a standard technology of Markov chains (mentioned, for example, in [5], it is easy to test the assumption that limit probabilities for M exist and are established by the equality

$$\pi_\sigma = \left(\begin{array}{c} w \\ \sigma_1 \dots \sigma_m \end{array} \right) \prod_{i=1}^m P^{\sigma_i}(a_i), \quad \sigma \in S$$

which proves Theorem 1.

Proof of Theorem 2.

Let us introduce a new scheme — the sliding window with random removing of elements (SWRRE). In this scheme a sequence of w "boxes" ($w \geq 1$) is considered. Every "box" may contain a letter from the alphabet A. As above, a Bernoulli source is given, generating the sequence $x_1 x_2 \dots, x_j \in A$ for all j, and let $P(a)$ be the probability of generating the letter $a \in A$.

In the initial moment there are letters from A in the "boxes". At every moment $t = 1, 2, \dots$ two operations are made: a random value ν^t is produced which is equal to $1, 2, \dots w$ with the probability $1/w$ each, and the letter from the box number ν^t is removed. Then a value φ^t is produced which is equal to $1, 2, \dots m$ such as

$$P\{\varphi^t = k\} = P(a_k)$$

and the letter a is located in the box which became free.

It is easy to see that the scheme SWRRE is an exact but more detailed model of the scheme ISW. In fact, denote by ν_k^t a random value which is equal to the number of boxes containing the letter a_k at the moment t and let $\nu^t = (\nu_1^t, \dots, \nu_m^t)$. It follows from the scheme SWRRE described that the probabilities of conversion from ν^t to ν^{t+1} are also defined by the equality (16). Hence, if the initial distribution is the same for both schemes ISW and SWRRE (i.e. $\bar{\nu}^0 = \nu^0$), then the distribution of probabilities for all other moments will be the same: for any $\bar{n} = (n_1, \dots, n_m)$

$$P\{\nu^t = \bar{n}\} = P\{\bar{\nu}^t = n\} \tag{17}$$

Then let us introduce a new random value φ^t which is connected with SWRRE. By definition, φ^t is equal to the number of boxes from which letters were not removed at the moments $1, 2, \dots, t$. Note immediately that the distribution of this value is well known (see, for example, [5]), when φ^t is the number of empty boxes obtained after the random distribution of t elements in w boxes). It is known that

$$P\left\{\varphi^t = 0\right\} = \sum_{k=0}^{w} (-1)^k \left(\begin{array}{c} w \\ k \end{array} \right) \left(1 - \frac{k}{w}\right)^t \tag{18}$$

(see [5]).

Let in the scheme of SWRRE all letters be replaced in all boxes (i.e. $\varphi^t = 0$) at time t. Then, obviously, the distribution of the vector ν^t does not depend on t and it is subjected to the multinomial distribution:

$$P\left\{\nu^t = n_1, \dots, \nu_m^t = n_m / \varphi^t = 0\right\} = \left(\begin{array}{c} w \\ n_1 \dots n_m \end{array} \right) \prod_{j=1}^{m} P(a_j)^{n_j}$$

That yields, along with (4),

$$P\{\bar{\nu}^\infty = \bar{n}\} = P\{\nu^t = \bar{n}/\varphi^t = 0\}$$

It follows from this that

$$P\{\nu^t = \bar{n}\} \geq P\{\nu^\infty = \bar{n}\} \; P\{\varphi^t = 0\}$$

From this and (17) we have

$$P\{\bar{\nu}^t = \bar{n}\} \geq P\{\nu^\infty\} \; P\{\varphi^t = 0\}$$

That yields, along with the definition of R^t (5),

$$R^t \leq \sum_{\bar{n} \in S} P\left\{\bar{\nu}^\infty = \bar{n}\right\} \log \frac{P\{\bar{\nu}^\infty = \bar{n}\}}{P\{\bar{\nu}^\infty = \bar{n}\} \; P\{\varphi^t = 0\}} =$$

$$- \sum_{\bar{\nu} \in S} P\left\{\bar{\nu}^\infty = \bar{n}\right\} \log P\{\varphi^t = 0\}$$

From this and (18) we obtain (6).

The Theorem is proved.

The proof of the Corollary readily follows from the known estimates of the number of empty boxes (see [5]).

The proof of Theorem 3. Denote by π^t the probability that contents of some definite box did not transform at the moments $1, 2, \dots t$. Then, it is easy to see that for any box,

$$\pi^t = (1 - 1/w)^t \tag{19}$$

Let us fix some letter $a_i \in A$ and define the random value Θ_k^t which is equal to 1 in the case if at the moment t the k box contains $a_i, k = 0, \dots, w$. Then it is easy to see that $E(\Theta_k^t) = (1 - \pi^t) \; P(a_i) + \pi^t \cdot E(\Theta_k^0)$ and

$$\nu_j^t = \sum_{k=1}^{j} \Theta_k^t$$

From the latter equalities we obtain that

$$E(\nu_i^t) = w(1 - \pi^t) \; P(a_i) + w \; \pi^t E(\Theta_k^0) \tag{20}$$

From the obvious inequality $0 \leq E(\Theta_k^0) \leq 1$ and from (20) we have

$$w(1 - \pi^t) \; P(a_i) \leq E(\nu_i^t) \leq w(1 - \pi^t) P(a_i) + w \; \pi^t$$

From this, we obtain

$$-w \; \pi^t \leq E(\bar{\nu}_i^t) - P(a_i) w \leq w \; \pi^t$$

That yields, along with (19) and the known inequality $(1 - \varepsilon) < e^{-\varepsilon}$ the conclusion of the Theorem 3.

References

1. *Aho A.V., Hopcroft J.E., Ullman J.D.* Data Structures and Algorithms, Addison-Wesley, Rading, MA, 1983.

2. *Bell T.C., Cleary J.G., Witten I.N.* Text compression, Prentice Hall, Inc., 1990.

3. *Bently J.L., Sleator D.D., Tarjan R.E., Wei V.K.* A Locally Adaptive Data Compression Scheme Comm. ACM, v.29, 1986, pp.320-330.

4. *Elias P.* Interval and Recency Rank Source Coding: Two On-Line Adaptive Variable-Length Schemes, IEEE Trans. Inform. Theory, v.33, N 1,1987, pp.3-10.

5. *Feller W.* An Introduction to Probability Theory and Its Applications, Jon Wiley Sons, New York, vol.1, second edition, 1970.

6. *Kendall M.G., Stuart A.* The Advanced Theory of Statistics, v.2. (Inference and Relationship), Charles Griffin and Co.Limited, 1966.

7. *Knuth D.E.* Dynamic Huffman coding, J.Algorithms, v.6, 1985, pp.163-180.

8. *Krichevsky R.* Universal Compression and Retrieval. Kluwer Academic Publishers, 1994.

9. *Murrey Sherk.* Self-Adjusting k-ary Search Tress, J.Algorithms, v.19, N 1, 1995, pp.25-44.

10. *Rissanen J.* Complexity of Strings in the Class of Markov Sources". IEEE Trans. Inform. Theory, v.32, N 4, 1986, pp.526-532.

11. *Ryabko B.Ya.* Information Compression by a Book Stack, Problems of Information Transmission, v.16, N 4, 1980, pp.16-21, (in Russian).

12. *Ryabko B.Ya.* A locally adaptive data compression scheme (Letter), Comm. ACM, v.30, N 9, 1987, p.792.

13. *Ryabko B.Ya.* A Fast On-Line Adaptive Code, IEEE Trans. Inform. Theory, v.38, N 4, 1992, pp.1400-1404.

14. *Ryabko B.Ya.* The complexity and Effetiveness of Prediction Algorithms, J. of Complexity, v. 10, 1994, pp. 281-295.

15. *Viller J.S., Krishnan P.* Optimal Prefetching via Data Compression // Journal of the ACM, v.43, N 5, 1996, pp.771-793.

16. *Ziv J., Lempel A.* A univercal algorithm for sequential data compression, IEEE Trans. Inform. Theory, v.23, N 3, 1977, pp.337-343.