Description of Binary Sequences Based on the Interval Linear Complexity Profile

Vladimir B. Balakirsky

Euler Institute for Discrete Mathematics and its Applications (EIDMA)
c/o Eindhoven University of Technology
P.O.Box 513, 5600 MB Eindhoven, The Netherlands

Abstract. We consider the problem of partial approximation of binary sequences by the outputs of linear feedback shift registers. A generalization of the linear complexity profiles of binary sequences leads to a sequence that is regarded as the profile of interval linear complexity. This concept is useful if we want to identify all fragments of the given binary sequence having length at least $2m$ which can be described by $m+n$ bits, where $n < m$. We show that a fixed j–th component belongs to such a fragment with probability $\sim 1/3$ in the ensemble of fair coin–tossing sequences. We also apply the interval linear complexity approach to identify corrupted positions of the sequence generated by a linear feedback shift register with known feedback polynomial, where “corruption” of the sequence means either deletion, insertion or inversion of its components.

1 Introduction

There is a class of problems, relevant to decoding, cryptology, and data compression, where we are interested in a proper partitioning of a given binary sequence into different fragments. The goal of this partitioning is to represent some fragments as the output sequences of different linear feedback shift registers, and we speak about partial linear approximations of the given sequence in this sense. The problems of this class can be effectively attacked using a generalized version of the linear complexity profile of the sequence [1], [2], which was introduced in [3] as the interval linear complexity profile. We will present some basic properties of the interval linear complexity and apply this notion to attack some problems related to effective description of binary sequences.

We will use the following notations. Given a binary sequence $(u_1, \ldots, u_N)$, we denote $\mathbf{u}|_i^j \triangleq (u_i, \ldots, u_j)$ for all i and j such that $1 \le i \le j \le N$.

Definition 1. *Let the integers j and n be fixed in such a way that $j \ge n+1$. A binary sequence $\mathbf{u}|_{n+1}^j$ is generated by the polynomial $f(z) \triangleq 1 + f_1 z + \ldots + f_n z^n$ with $f_1, \ldots, f_n \in \{0,1\}$ given $\mathbf{u}|_1^n$ if and only if*

$$u_i = f_1 u_{i-1} \oplus \ldots \oplus f_n u_{i-n}, \quad \textit{for all} \;\; i = n+1, \ldots, j.$$

2 Some Properties of the Linear Complexity of Binary Sequences

Definition 2. *The linear complexity of the sequence* $\mathbf{u}|_1^j$, *denoted by* λ_j, *is the smallest integer* n *such that there exists a polynomial*

$$f_j(z) \triangleq 1 + f_{j,1}z + \ldots + f_{j,n-1}z^{n-1} + z^n$$

with $f_{j,1}, \ldots, f_{j,n-1} \in \{0,1\}$ *generating* $\mathbf{u}|_{n+1}^j$ *given* $\mathbf{u}|_1^n$. *The sequence* $(\lambda_1, \ldots, \lambda_N)$ *is referred to as the linear complexity profile* (*LCP*) *of* $\mathbf{u}|_1^N$.

The LCP of the sequence $\mathbf{u}|_1^N$ can be constructed by the Berlekamp–Massey (BM) algorithm [2]. We present this algorithm to introduce notations needed for further extensions.

1. Set $j = 1$, $(\lambda_0, f_0(z)) = (\lambda, f(z)) = (\lambda', f'(z)) = (0, 1)$, and $\delta = 0$.
2. Increase δ by 1, set

$$\begin{aligned}
\hat{u}_j &= \begin{cases} f_1 u_{j-1} \oplus \ldots \oplus f_\lambda u_{j-\lambda}, & \text{if } \lambda > 0 \\ 0, & \text{if } \lambda = 0 \end{cases} \\
\delta_j &= \delta \\
(\lambda'_j, f'_j(z)) &= (\lambda', f'(z)) \\
(\lambda_j, f_j(z)) &= \begin{cases} (\lambda, f(z)), & \text{if } u_j = \hat{u}_j \\ (\max\{\lambda, \lambda' + \delta\}, f(z) + z^\delta f'(z)), & \text{if } u_j \neq \hat{u}_j. \end{cases}
\end{aligned}$$

3. If $\lambda_j > \lambda$, then set $(\lambda', f'(z)) = (\lambda, f(z))$, and $\delta = 0$.
4. Set $(\lambda, f(z)) = (\lambda_j, f_j(z))$.
5. Increase j by 1 and if $j \leq N$, then go to 2.
6. End.

Let $0 = \Lambda_0 < \Lambda_1 < \ldots < \Lambda_T$ be the values of the LCP of the sequence $\mathbf{u}|_1^N$ defined by (see Fig.1)

$$j \in \{0, \ldots, N\},\ \lambda_j = j/2 \iff j \in \{2\Lambda_0, 2\Lambda_1, \ldots, 2\Lambda_T\}. \tag{1}$$

For all $t = 0, \ldots, T$, let

$$F_t(z) \triangleq f_{2\Lambda_t}(z),\quad D_t \triangleq \deg F_t(z). \tag{2}$$

Lemma 1 ([1]). *For any binary sequence* $\mathbf{u}|_1^N$,

$$\begin{aligned}
j \in [\Lambda_{t-1} + \Lambda_t, \Lambda_t + \Lambda_{t+1}) &\Longrightarrow \lambda_j = \Lambda_t \\
j \in [2\Lambda_t, \Lambda_i + \Lambda_{t+1}) &\Longrightarrow f_j(z) = F_t(z).
\end{aligned}$$

Furthermore, for all $t = 1, \ldots, T$, *the polynomial* $F_t(z)$ *is uniquely determined by the vector* $(u_1, \ldots, u_{2\Lambda_t})$.

Lemma 2. *For any binary sequence* $\mathbf{u}|_1^N$,

(a) $D_{t-1} = \Lambda_{t-1},\ D_t < \Lambda_t \implies D_{t+1} = \Lambda_{t+1}$;

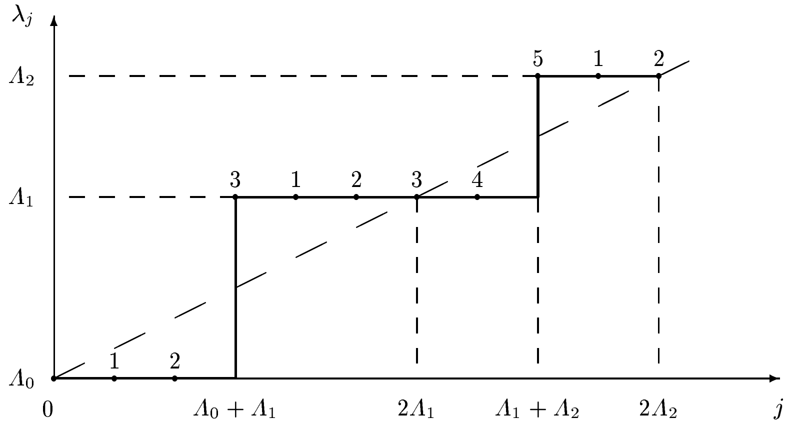

Fig. 1. The LCP of a binary sequence and the values $\delta_j, j = 1, 2, \ldots$

(b) $D_{t-1} = \Lambda_{t-1},\ D_t = \Lambda_t \implies D_{t+1} = \Lambda_{t+1}$ *or* $D_{t+1} < \Lambda_{t+1}$ *depending on* $u_{2\Lambda_{t+1}}$;
(c) *there are no binary sequences with* $D_{t-1} < \Lambda_{t-1}$ *and* $D_t < \Lambda_t$

for all $t = 1, \ldots, T$, *where* $D_{-1} \triangleq \Lambda_{-1} \triangleq 0$.

Proof : The polynomial $f_j(z)$, constructed by the BM algorithm, can be expressed as

$$f_j(z) = \begin{cases} f_{j-1}(z), & \text{if } u_j = \hat{u}_j \\ f_{j-1}(z) + z^{\delta_j} f_j'(z), & \text{if } u_j \neq \hat{u}_j. \end{cases}$$

Let us fix $t \in \{1, \ldots, T\}$ and note that

$$j \in (\Lambda_{t-1} + \Lambda_t, \Lambda_t + \Lambda_{t+1}] \implies \begin{cases} \delta_j = j - (\Lambda_{t-1} + \Lambda_t) \\ f_j'(z) = F_{t-1}(z). \end{cases} \tag{3}$$

Denote $i = \Lambda_t + \Lambda_{t+1}$. Then $f_i(z) = f_{i-1}(z) + z^{\delta_i} f_i'(z)$, where $\delta_i = \Lambda_{t+1} - \Lambda_{t-1}$ and $f_i'(z) = F_{t-1}(z)$. Therefore

$$\deg[z^{\delta_i} f_i'(z)] = \Lambda_{t+1} - \Lambda_{t-1} + D_{t-1}$$

and since $\deg f_{i-1}(z) \leq \Lambda_t < \Lambda_{t+1}$, we have

$$D_{t-1} = \Lambda_{t-1} \implies \deg f_i(z) = \Lambda_{t+1}.$$

Similarly to (3),

$$j \in (\Lambda_t + \Lambda_{t+1}, 2\Lambda_{t+1}] \implies \begin{cases} \delta_j = j - (\Lambda_t + \Lambda_{t+1}) \\ f_j'(z) = F_t(z) \end{cases}$$

Thus,

$$\deg[z^{\delta_j} f_j'(z)] = j - (\Lambda_t + \Lambda_{t+1}) + D_t = j - \Lambda_{t+1} - (\Lambda_t - D_t)$$

and

$$j \in (\Lambda_t + \Lambda_{t+1}, 2\Lambda_{t+1}) \implies j - \Lambda_{t+1} - (\Lambda_t - D_t) < \Lambda_{t+1}.$$

As a result,

$$j \in (\Lambda_t + \Lambda_{t+1}, 2\Lambda_{t+1}) \implies \deg f_j(z) = \Lambda_{t+1}.$$

If $D_t < \Lambda_t$, then this equality is also valid for $j = 2\Lambda_{t+1}$. Otherwise, if $D_t = \Lambda_t$, then $D_{t+1} = \Lambda_{t+1}$ for $u_j = \hat{u}_j$ and $D_{t+1} < \Lambda_{t+1}$ for $u_j \neq \hat{u}_j$.

These considerations prove first two statements. Suppose the third conclusion is false and let t be the least value so that $D_t < \Lambda_t$ and $D_{t+1} < \Lambda_{t+1}$. Then $t > 0$ since $D_0 = \Lambda_0$, so $D_{t-1} = \Lambda_{t-1}$. But then conclusion (a) of the lemma implies that $D_{t+1} = \Lambda_{t+1}$, a contradiction.

Lemma 3. *Let $K_{2n}^{(i)}$ be the number of binary sequences $\mathbf{u}|_1^{2n}$ such that*

$$u_1 = \ldots = u_i = 0, \ \ u_{i+1} = 1 \tag{4}$$

and

$$\lambda_{2n} = \deg f_{2n}(z) = n. \tag{5}$$

Then, for all $n > i$,

$$K_{2n}^{(i)} = \frac{1}{6} \cdot 2^{2n-i} + \frac{1}{3} \cdot 2^i. \tag{6}$$

Proof : Let $K_{2n}^{(i)\prime}$ be the number of binary sequences of length $2n$ satisfying (4) such that $\lambda_{2n} = n$ and $\deg f_{2n}(z) < n$. Let us consider binary sequences $\mathbf{u}|_1^{2n}$ satisfying (4) whose LCP $\lambda_1, \lambda_2, \ldots$ crosses the line with the slope $1/2$ in the neighboring points $2n - 2l$ and $2n$, i.e., $\lambda_{2n-2l} = n - l$, $\lambda_{2n} = n$, and $\lambda_j \neq j/2$ for all $j = 2n - 2l + 1, \ldots, 2n - 1$. In other words, $\Lambda_t = n - l$ and $\Lambda_{t+1} = n$ for some index t in the sequence of even indices j with the property $\lambda_{j/2} = j$. Then $\lambda_{(2n+2n-2l)/2} = \lambda_{2n-l} = n$ by Lemma 1 and this equation is valid if we fix only first $2n - l$ bits of the sequence $\mathbf{u}|_1^{2n}$ in a certain way : for any $\mathbf{u}|_{2n-l+1}^{2n}$, the LCP of $\mathbf{u}|_1^{2n}$ at position $2n$ is equal to n. However, the degree of the feedback polynomial at position $2n$ (this polynomial is unique by Lemma 1) can be either less than n or equal to n. If $\mathbf{u}|_1^{2n-l}$ is fixed in such a way that $D_t < \Lambda_t$, then these l bits can be chosen in arbitrary way and $D_{t+1} = n$ (statements (a), (c) of Lemma 2). If $\mathbf{u}|_1^{2n-l}$ is fixed in such a way that $D_t = \Lambda_t$, then, for any $l - 1$ bits $u_{2n-l+1}, \ldots, u_{2n-1}$, there exists exactly one assignment for u_{2n} such that $D_{t+1} = n$ and exactly one assignment such that $D_{t+1} < n$ (statement (b) of Lemma 2). Thus, the contribution of sequences $\mathbf{u}|_1^{2n}$ under considerations to $K_{2n}^{(i)}$ is $K_{2n-2l}^{(i)} 2^{l-1} + K_{2n-2l}^{(i)\prime} 2^l$ and the contribution of these sequences to

$K_{2n}^{(i)\prime}$ is $K_{2n-2l}^{(i)} 2^{l-1}$. We take the sum of these quantities on $l = 1, \ldots, n-1$ and obtain

$$K_{2n}^{(i)} = \frac{1}{2} \sum_{l=1}^{n-i-1} K_{2n-2l}^{(i)} 2^l + \sum_{l=1}^{n-i-1} K_{2n-2l}^{(i)\prime} 2^l \tag{7}$$

$$K_{2n}^{(i)\prime} = \frac{1}{2} \sum_{l=1}^{n-i-1} K_{2n-2l}^{(i)} 2^l. \tag{8}$$

Furthermore, since $\lambda_{2i+2} = i+1$ and $\lambda_{2t} \neq t$ for all $t = 1, \ldots, 2i+1$,

$$K_{2i+2}^{(i)} = K_{2i+2}^{(i)\prime} = 2^i. \tag{9}$$

Substituting (8) in (7) and using (9) we have

$$\begin{aligned}
K_{2n}^{(i)} &= \frac{1}{2} \sum_{l=1}^{n-i-1} K_{2n-2l}^{(i)} 2^l + \sum_{l=1}^{n-i-2} \frac{1}{2} \sum_{l'=1}^{n-l-i-1} K_{2n-2l-2l'}^{(i)} 2^{l+l'} \\
&\quad + K_{2i+2}^{(i)\prime} 2^{n-i-1} \\
&= \frac{1}{2} \sum_{l=1}^{n-i-1} K_{2n-2l}^{(i)} 2^l + \frac{1}{2} \sum_{l=1}^{n-i-1} (l-1) K_{2n-2l}^{(i)} 2^l + 2^{n-1} \\
&= \frac{1}{2} \sum_{l=1}^{n-i-1} l K_{2n-2l}^{(i)} 2^l + 2^{n-1}
\end{aligned}$$

After substitution of the expression on the right–hand side of (6) for $K_0^{(i)}$, $K_2^{(i)}, \ldots$ and simple manipulations, we come to the identity, as it can be proven from the identities

$$\sum_{l=1}^{n-i-1} l 2^{-l} = 2(1 - (n-i+1) 2^{-n+i})$$

and

$$\sum_{l=1}^{n-i-1} l 2^{l} = 2(1 + (n-i-2) 2^{n-i-1}).$$

Hence, if (6) is valid, then both (7), (8), and (9) are satisfied.

3 The m–Interval Linear Complexity of Binary Sequences

In the following considerations, we denote $\mathbf{u}_j \triangleq (u_j, \ldots, u_{j-m+1})$, $j = 1, \ldots, N$, where $u_0 = \ldots = u_{-m} = 0$, for a fixed binary sequence $\mathbf{u}|_1^N$ and integer $m \geq 1$. Furthermore, let $\mathbf{0}$ denote the all–zero vector of length m.

Definition 3. *Given an integer $m \geq 1$ and a binary sequence $\mathbf{u}|_1^N$ with $u_1 = 1$, the value of the m–interval linear complexity at position $j \in \{m+1, \ldots, N\}$, denoted by μ_j, is defined by the following rules :*

- ⊙ *if $\mathbf{u}_j = \mathbf{0}$, then $\mu_j = 0$;*
- ⊙ *if $\mathbf{u}_j \neq \mathbf{0}$, then μ_j is the smallest integer n such that the vector $\mathbf{u}_j$ can be expressed as a linear combination of the vectors $\mathbf{u}_{j-1}, \ldots, \mathbf{u}_{j-n}$, i.e., there exists a polynomial*

$$g_j(z) \stackrel{\triangle}{=} 1 + g_{j,1}z + \ldots + g_{j,n-1}z^{n-1} + z^n$$

with $g_{j,1}, \ldots, g_{j,n-1} \in \{0, 1\}$ and

$$\mathbf{u}_j = g_{j,1}\mathbf{u}_{j-1} \oplus \ldots \oplus g_{j,n-1}\mathbf{u}_{j-n+1} \oplus \mathbf{u}_{j-n}.$$

The sequence $(\mu_{m+1}, \ldots, \mu_N)$ is referred to as the m–interval linear complexity profile (m–LCP) of $\mathbf{u}|_1^N$.

If $\mathbf{u}_j \neq \mathbf{0}$, then we can construct the pair $(\mu_j, g_j(z))$ using the following variant of the BM algorithm (see Table 1 for the illustration).

1′. Set $t = 0$, $(\lambda, f(z)) = (\lambda', f'(z)) = (0, 1)$, and $\delta = 0$.
2′. Increase δ by 1, set

$$\begin{aligned}
\hat{u}_{j-t} &= \begin{cases} f_1 u_{j-t+1} \oplus \ldots \oplus f_\lambda u_{j-t+\lambda}, & \text{if } \lambda > 0 \\ 0, & \text{if } \lambda = 0 \end{cases} \\
\delta_t &= \delta \\
(\lambda'_t, f'_t(z)) &= (\lambda', f'(z)) \\
(\lambda_t, f_t(z)) &= \begin{cases} (\lambda, f(z)), & \text{if } \ u_{j-t} = \hat{u}_{j-t} \\ (\max\{\lambda, \lambda' + \delta\}, f(z) + z^\delta f'(z)), & \text{if } \ u_{j-t} \neq \hat{u}_{j-t}. \end{cases}
\end{aligned}$$

3′. If $\lambda_t > \lambda$, then set $(\lambda', f'(z)) = (\lambda, f(z))$, and $\delta = 0$.
4′. Set $(\lambda, f(z)) = (\lambda_t, f_t(z))$.
5′. If $t - \lambda < m$ or $\deg f(z) \neq \lambda$, then increase t by 1 and go to 2′.
6′. Set $(\mu_j, g_j(z)) = (\lambda, f(z))$. End.

Some properties of the m–LCP of a binary sequence are summarized below and illustrated in Fig. 2 and Fig. 3.

Theorem 1. *For all $j \geq m+1$ and $n \in \{0, \ldots, m-1\}$,*

$$\left.\begin{array}{l} \mu_{j-1} \neq n \\ \mu_j = \ldots = \mu_{j+l-1} = n \\ \mu_{j+l} \neq n \end{array}\right\} \Longrightarrow \left\{\begin{array}{l} \mu_{j-1} \geq m \\ g_j(z) = \ldots = g_{j+l-1}(z) \\ g_j(z) \text{ is unique} \\ \mu_{j+l} = \ldots = \mu_{j+l+h-1} = m + l > m \\ \mu_{j+l+h} \leq m \end{array}\right.$$

where $h \stackrel{\triangle}{=} m - n$. Furthermore, if $j > 2m$ and $u_2, \ldots, u_j$ are i.i.d. random variables chosen from $\{0, 1\}$ with probability $1/2$, then

$$\Pr\{\, \mu_j = 0 \,\} = \frac{1}{2^m} \tag{10}$$

$$\Pr\{\, \mu_j = n \,\} = \frac{2^n - 2^{-n}}{3 \cdot 2^m}, \quad \text{for all } \ n = 1, \ldots, m \tag{11}$$

Table 1. Constructing of the pair $(\mu_j, g_j(z))$ for $j = 13$ and the sequence $\mathbf{u}|_1^{13} = (1000010111001)$ when $m = 5$.

t	u_{j-t}	$\hat{u}_{j-t}$	λ_t	$f_t(z)$	λ'_t	$f'_t(z)$	δ_t
0	1	0	1	$1+z$	0	1	0
1	0	1	1	1	0	1	1
2	0	0	1	1	0	1	2
3	1	0	3	$1+z^3$	1	1	0
4	1	0	3	$1+z^2+z^3$	1	1	1
5	1	1	3	$1+z^2+z^3$	1	1	2
6	0	0	3	$1+z^2+z^3$	1	1	3
7	1	1	3	$1+z^2+z^3$	1	1	4

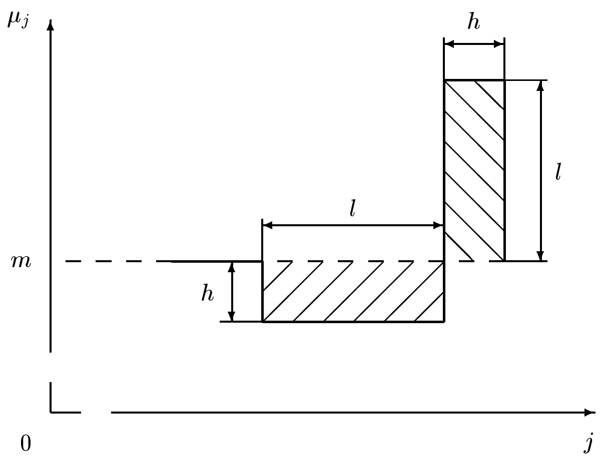

Fig. 2. Fragment of the m–LCP of any binary sequence.

and

$$\Big(\Pr\{\,\mu_j < m\,\}, \Pr\{\,\mu_j = m\,\}, \Pr\{\,\mu_j > m\,\}\Big)$$
$$= \Big(\frac{1}{3} + \frac{2}{3}\cdot 2^{-2m}, \ \frac{1}{3} - \frac{1}{3}\cdot 2^{-2m}, \ \frac{1}{3} - \frac{1}{3}\cdot 2^{-2m}\Big). \qquad (12)$$

Proof : Suppose that the sequence $\mathbf{u}|_{j-m+1}^{j}$ is generated by a feedback polynomial of degree $n = \mu_j$ given $\mathbf{u}|_{j-m-n+1}^{j-m}$. Then the sequence $u_{j-m+1-n}, \ldots, u_j$ of length $m+n$ can be specified by $2n$ bits : n bits determine coefficients of the feedback polynomial and n bits determine the ini-

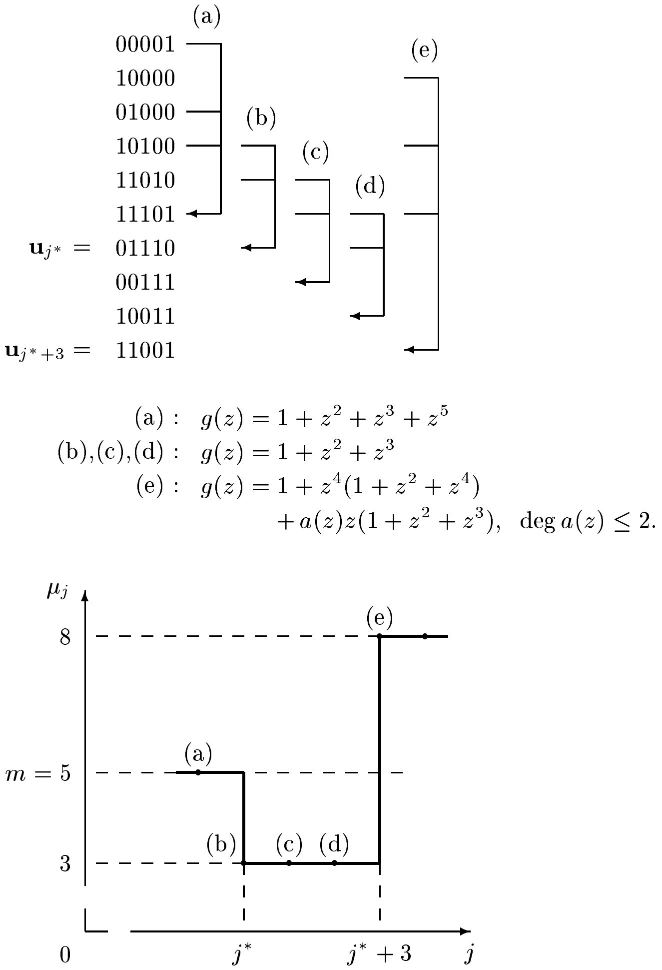

Fig. 3. Fragment of the 5–LCP of the sequence $(u_{j^*-10}, u_{j^*-9}, \ldots) = 10000101110011\ldots$

tial content of the shift register. All other bits are computed or "generated". Thus, we know that the sequence $u_{j-m+1-n}, \ldots, u_{j-1}$ of length $m+n-1$ can be also specified by $2n$ bits and, since $\mu_j = n < m$, the feedback polynomial of degree n is unique. If we take this sequence in the reverse order, i.e., consider the sequence $u_{j-1}, u_{j-2}, \ldots, u_{j-m+1-n}$, then the same conclusion (the sequence $u_{j-n+1}, \ldots, u_{j-m+1-n}$ is generated by a polynomial of degree n given $u_{j-1}, \ldots, u_{j-n}$) is valid. If we can compute u_{j-m-n} using this polynomial, then $\mu_{j-1} = n$, a contradiction. Otherwise, the linear complexity of

$u_{j-1}, u_{j-2}, \ldots, u_{j-m-n}$ at the last position is at least m (Lemma 1), and the statement : "$\mu_{j-1} \neq n, \mu_j = n < m$ implies $\mu_{j-1} \geq m$" follows.

Similar considerations lead to the conclusion that there exists only one polynomial $g_j(z)$ generating the sequence $\mathbf{u}|_{j-m+1}^{j+l-1}$ given $\mathbf{u}|_{j-m-n}^{j-m}$.

Let us fix $\Delta l \in \{0, \ldots, h-1\}$ and denote $t = j + l + \Delta l$. Then the vector $\mathbf{u}_t$ can be expressed as a linear combination of the vectors $\mathbf{u}_{t-1}, \ldots, \mathbf{u}_{t-\mu_t-1}$ and μ_t is the smallest integer with this property. On the other hand, $\mathbf{u}_j$ can be expressed as a linear combination of the vectors $\mathbf{u}_{j-1}, \ldots, \mathbf{u}_{j-n}$. Therefore $j - n > t - \mu_t$, which implies $\mu_t > n + l + \Delta l$. Thus, $\hat{\mathbf{u}} = \mathbf{u}_{j-m-n+1}^{j+l}$ is a fragment of the sequence $\mathbf{u}_{j-m-\mu_t+1}^{t}$ for all $t = j+l, \ldots, j+l+h-1$. The length of the sequence $(\hat{\mathbf{u}}, u_{j+l}, \ldots, u_{j+l+h-1})$ is equal to $2m + l - 1$ and its linear complexity at position $m + n + l - 1$ is equal to n. Thus, the linear complexity at positions $m + n + l, \ldots, 2m + l - 1$ is equal to $m + l$ (Lemma 1). As a result, inequality $\mu_t > n + l + \Delta l$ can be replaced with the stronger inequality $\mu_t \geq m + l$, for all $t = j+l, \ldots, j+l+h-1$. These inequalities are tight since the matrix consisting of rows $\mathbf{u}_{t-m-l}, \ldots, \mathbf{u}_{t-1}$ has rank m for all $t = j+l, \ldots, j+l+h-1$.

The inequality $\mu_{j+l+m-n} \leq m$ follows from the fact that the vectors $\mathbf{u}_{j+l+m-n-1}, \ldots, \mathbf{u}_{j+l-n-1}$ are linearly independent.

Equality (10) holds since $\mu_j = 0$ if and only if $\mathbf{u}|_{j-m+1}^{j} = \mathbf{0}$.

For all $n = 1, \ldots, m$, we have

$$\Pr\{\, \mu_j = n \,\} = \frac{1}{2^{m+n}} \sum_{i=0}^{n-1} K_{2n}^{(i)}$$

since the sum on the right–hand side is the number of sequences $\mathbf{u}|_{j-m-n+1}^{j}$ with $\mu_j = n$. Thus, (11) follows from (6) after simple computations, and (12) follows from

$$\Pr\{\, \mu_j < m \,\} = \Pr\{\, \mu_j = 0 \,\} + \sum_{n=1}^{m-1} \Pr\{\, \mu_j = n \,\}.$$

Remark. If $\mu_j = \ldots = \mu_{j+l-1} = n$, then the sequence $\mathbf{u}|_{j-m+1}^{j+l-1}$ is generated at the output of the linear feedback shift register having the feedback polynomial $g_j(z)$ of length n if we load $\mathbf{u}|_{j-m-n+1}^{j-m}$ in the register. This sequence can be defined by the n coefficients of the polynomial $g_j(z)$ instead of $m + l - 1$ bits provided that j is fixed and the initial contents of the shift register is known. Hence, $m - n + l - 1$ is the number of bits, which can be compressed if we consider only this fragment. This number can be viewed as "a potential redundancy" in the sense that the real compression is attained only if we also know n and l. Nevertheless, the fact that the probability that $\mu_j < m$ is close to 1/3 can be useful to characterize closeness of a given sequence to "a fair coin–tossing sequence".

4 Identification of the Positions Where the m–Interval Linear Complexity is Less Than m

Lemma 3. *Given an $i \geq m+1$, let $\mathbf{C}_i$ and $\mathbf{X}_i$ be binary $m \times m$ matrices containing rows $\mathbf{c}_{i,k} = (c_{i,k}^{(1)}, \ldots, c_{i,k}^{(m)})$ and $\mathbf{x}_{i,k} = (x_{i,k}^{(1)}, \ldots, x_{i,k}^{(m)})$, respectively, $k = 1, \ldots, m$. Suppose that the given vector $\mathbf{u}_j \in \{0,1\}^m$ can be represented as*

$$\mathbf{u}_j = f_1 \mathbf{x}_{i,1} \oplus f_2 \mathbf{x}_{i,2} \oplus \ldots \oplus f_m \mathbf{x}_{i,m} \tag{13}$$

where $f_2, \ldots, f_m \in \{0,1\}$ and $f_1 = 1$. Define the rows of the matrices $\mathbf{C}_j$ and $\mathbf{X}_j$ by

$$\mathbf{c}_{j,k} \triangleq \mathrm{L}[\,\mathbf{c}_{i,k}\,] \oplus \begin{cases} \mathbf{0}, & \text{if } c_{i,k}^{(1)} = 0 \\ (f_2, \ldots, f_{m-1}, 1), & \text{if } c_{i,k}^{(1)} = 1 \end{cases} \tag{14}$$

$$\mathbf{x}_{j,k} \triangleq \begin{cases} \mathbf{x}_{i,k+1}, & \text{if } k \in \{1, \ldots, m-1\} \\ \mathbf{u}_j, & \text{if } k = m \end{cases} \tag{15}$$

where $\mathrm{L}[\,\mathbf{c}_{i,k}\,] = (c_{i,k}^{(2)}, \ldots, c_{i,k}^{(m)}, 0)$ and $k = 1, \ldots, m$. Then

$$\mathbf{C}_j \mathbf{X}_j = \mathbf{C}_i \mathbf{X}_i. \tag{16}$$

Proof : By (13) and $f_1 = 1$,

$$x_{i,1}^{(n)} = f_2 x_{i,2}^{(n)} \oplus \ldots \oplus f_m x_{i,m}^{(n)} \oplus u_{j-n+1}$$

for all $n = 1, \ldots, m$. Using this equation and (14), (15) we express the (k,n)–th entry of the matrix $\mathbf{C}_j \mathbf{X}_j$ as

$$\begin{aligned} & c_{i,k}^{(1)} x_{i,1}^{(n)} \oplus \ldots \oplus c_{i,k}^{(m)} x_{i,m}^{(n)} \\ &= (c_{i,k}^{(2)} \oplus f_2 c_{i,k}^{(1)}) x_{i,2}^{(n)} \oplus \ldots \oplus (c_{i,k}^{(m)} \oplus f_m c_{i,k}^{(1)}) x_{i,m}^{(n)} \oplus c_{i,k}^{(1)} u_{j-n+1} \\ &= c_{j,k}^{(1)} x_{j,1}^{(n)} \oplus \ldots \oplus c_{j,k}^{(m-1)} x_{j,m-1}^{(n)} \oplus c_{j,k}^{(m)} x_{j,m}^{(n)} \end{aligned}$$

and (16) follows.

Lemma 4. *If the rank of the matrix $\mathbf{X}_i$ is equal to m and the matrix $\mathbf{C}_i$ is determined by the equation*

$$\mathbf{C}_i \mathbf{X}_i = \mathbf{I} \tag{17}$$

where $\mathbf{I}$ denotes the $m \times m$ identity matrix, then the coefficients $f_1, \ldots, f_m$, satisfying (13) given $\mathbf{u}_j$, can be expressed as

$$(f_1, \ldots, f_m) = \mathbf{u}_j \mathbf{C}_i. \tag{18}$$

Proof : Using (17) we have $\mathbf{u}_j(\mathbf{C}_i \mathbf{X}_i) = \mathbf{u}_j$. However $\mathbf{u}_j(\mathbf{C}_i \mathbf{X}_i) = (\mathbf{u}_j \mathbf{C}_i)\mathbf{X}_i$, and (18) follows since the linear independence of rows of the matrix $\mathbf{X}_i$ implies the uniqueness of the solution to this equation.

Table 2. Example of the transformation of the basis for $i = 10, j = 18, \mathbf{u}_j = 10010$, $(f_1, \ldots, f_5) = \mathbf{u}_j \mathbf{C}_i = 10101$.

$\mathbf{C}_i$	$\mathbf{X}_i$	$\mathbf{e}_i$	$\mathbf{C}_j$	$\mathbf{X}_j$	$\mathbf{e}_j$
10011	10001	4	01101	11000	12
11011	11000	3	11101	11100	11
01100	11100	2	11000	11110	10
00110	11110	1	01100	11111	9
00011	11111	0	00110	10010	0

The transformation $(\mathbf{C}_i, \mathbf{X}_i) \to (\mathbf{C}_j, \mathbf{X}_j)$ is illustrated in Table 2.

Since $u_1 = 1$, the vectors $\mathbf{u}_1, \ldots, \mathbf{u}_m$ are linearly independent, and we can uniquely determine the matrix $\mathbf{C}_m$ by (17), where $\mathbf{x}_{m,k} = \mathbf{u}_k$, $k = 1, \ldots, m$. Let us also set $p_m = 0$ and run the following algorithm for all $j = m+1, \ldots, N$.

1^*. Find the largest $i \in \{m+1, \ldots, j-1\}$ such that $p_j \neq *$ and compute $(f_1, \ldots, f_m)$ by (18).
2^*. If $f_1 = 0$, then set $p_j = *$.
3^*. If $f_1 = 1$, then set $p_j = i$ and form the matrix $\mathbf{C}_j$ by (14).

The statement below directly follows from Theorem 1 and description of the algorithm above.

Theorem 2. *Let $(\mu_{m+1}, \ldots, \mu_N)$ be the m–LCP of $\mathbf{u}|_1^N$. If $p_{m+1}, \ldots, p_N$ are constructed by the $1^* - 3^*$ algorithm, then $p_j \neq *$ implies $\mu_j = m$ and $p_j = *$ implies $\mu_j < m$, for all $j = m+1, \ldots, N$.*

5 Identification of the Positions Where the m–Fragments of the Sequence, Generated by the Known Polynomial, are Corrupted

Suppose that the binary sequence $\mathbf{v}|_{m+1}^{N'} \triangleq (v_1, \ldots, v_{N'})$ is generated by the polynomial

$$g(z) \triangleq 1 + g_1 z + \ldots + g_{m-1} z^{m-1} + z^m$$

given $\mathbf{v}|_1^m$ i.e.,

$$v_j = g_1 v_{j-1} \oplus \ldots \oplus g_{m-1} v_{j-m+1} \oplus v_{j-m}, \quad \text{for all} \quad j = m+1, \ldots, N'. \tag{19}$$

Then the sequence $\mathbf{v}|_1^{N'}$ is defined by $2m-1$ bits $v_1, \ldots, v_m, g_1, \ldots, g_{m-1}$, and specification of these bits is the shortest possible description of $\mathbf{v}|_1^{N'}$ in most of the cases. Suppose that this sequence is distorted in such a way that some components are deleted, some 0's and 1's are inserted, and some components are inverted. We assume that the distorting process is implemented in accordance with the following algorithm that will be referred to as *the (D1)–(D4) procedure* (see Table 3 for the illustration).

Let $(\alpha_1, \ldots, \alpha_{N'})$ be a given vector whose components belong to the set $\{+, D, I_0, I_1, S\}$.

Set $i = j = 1$ and until $i \leq N'$ do

(*D1*) if $\alpha_i = +$, then : set $u_j = v_i$; increase i and j by 1;
(*D2*) if $\alpha_i = D$, then : increase i by 1;
(*D3*) if $\alpha_i = I_b$, where $b \in \{0, 1\}$, then : set $u_j = b$, $u_{j+1} = v_i$; increase i by 1 and j by 2;
(*D4*) if $\alpha_i = S$, then : set $u_j = v_i \oplus 1$; increase i and j by 1.

The value of integer j at the output of the algorithm is the length of the received sequence, i.e., $j = N$. Given $\mathbf{u}|_1^N$, we will construct two vectors, the vector of references $(r_1, \ldots, r_N)$ and the vector of shifts $(\Delta_1, \ldots, \Delta_N)$, in such a way that $r_j \in \{0, \ldots, N\} \bigcup \{-\}$, where $-$ is a special symbol, and $\Delta_1, \ldots, \Delta_N$ are integers. For all $i = 1, \ldots, N'$, we will denote $\mathbf{v}_i \triangleq (v_i, \ldots, v_{i-m+1})$ where $v_0 = \ldots = v_{-m} = 0$.

Lemma 6. *Suppose that components of the sequence $\mathbf{v}|_1^{N'}$ satisfy* (19) *and that the sequence $\mathbf{u}|_1^N$ is constructed from $\mathbf{v}|_1^{N'}$ by the (D1)–(D4) procedure. Suppose also that the vector of references and the vector of shifts are constructed in such a way that the following properties hold. Let $j \in \{m+1, \ldots, N\}$ be any index with $r_j \neq -$ and $r_{j_i} \neq -$ for all $i = 1, \ldots, m$, where $j_m = r_j, j_{m-1} = r_{j_m}, \ldots, j_1 = r_{j_2}$. Then*

(a) $j_1 < j_2 < \ldots < j_m < j$;
(b) $\mathbf{u}_{j_1}, \ldots, \mathbf{u}_{j_m}$ *are linearly independent vectors*;
(c) $\mathbf{u}_j = (f_1 \mathbf{u}_{j_1}) \oplus \ldots \oplus (f_m \mathbf{u}_{j_m})$;
(d) $f_1 = 1$;
(e) $\mathbf{u}_{j_i} = \mathbf{v}_{j_i + \Delta_{j_i}}$ *for all* $i = 1, \ldots, m$ *and* $\mathbf{u}_j = \mathbf{v}_{j+\Delta_j}$

where coefficients $f_1, \ldots, f_m$ are uniquely determined. If conditions (a)–(e) are satisfied, then

$$g(z) \;\Big|\; f_1 z^{e_1} + \ldots + f_m z^{e_m} + 1 \tag{20}$$

where $e_i \triangleq j + \Delta_j - (j_i + \Delta_i)$ for all $i = 1, \ldots, m$.

Proof : Denote $\mathcal{E} = \{e_i, i = 1, \ldots, m : \; f_i = 1\}$ and $\mathbf{g} = (g_0, g_1, \ldots, g_m)$, where $g_0 = g_m = 1$ and $g_1, \ldots, g_{m-1}$ are coefficients of the polynomial $g(z)$. Let us construct a binary vector $(F_0, \ldots, F_{e_1})$ by the following multi–step algorithm. Initially $(F_0, \ldots, F_m) = \mathbf{g}$ and $F_{m+1} = \ldots = F_{e_1} = 0$. At each step, find some $h \in \{1, \ldots, e_1 - m\} \backslash \mathcal{E}$ such that $F_h = 1$ and, if the search is successful, replace F_{h+k} with $F_{h+k} \oplus g_k$ for all $k = 0, \ldots, m$. Since components of the sequence $\mathbf{v}|_1^{N'}$ satisfy (19), $\sum_{h:F_h=1} \mathbf{v}_h = \mathbf{0}$ at all steps of the algorithm. Furthermore, by the definition of parameters $e_1, \ldots, e_m$,

$$\sum_{h:F_h=1} \mathbf{v}_h = \mathbf{0} \iff g(z) \Bigm| F_{e_1} z^{e_1} + F_{e_m} z^{e_m} + 1.$$

Conditions of the lemma imply that the algorithm terminates when $F_h = 1$ for all $h \in \mathcal{E}$ and $F_h = 0$ for all $h \in \{1, \ldots, e_1 - 1\} \backslash \mathcal{E}$. This statement is equivalent to (20).

For example, let us consider data in Table 3. Let $r_1^0, r_2^0, \ldots$ denote components of the vector of references and $\Delta_1^0, \Delta_2^0, \ldots$ denote components of the vector of shifts. If $j = 18$, then

$$r_{18}^0 = 10,\ r_{10}^0 = 9,\ r_9^0 = 8,\ r_8^0 = 7,\ r_7^0 = 6.$$

Thus $(j_1, \ldots, j_5) = (6, 7, 8, 9, 10)$. Conditions of Lemma 6 are satisfied with $(f_1, \ldots, f_5) = 10101$ and the matrix, which was denoted by $\mathbf{X}_j$ in Table 2. Furthermore, $r_j^0 + \Delta_j^0 = 19$ and $(\Delta_6^0, \ldots, \Delta_{10}^0) = (0, 0, 0, 0, 0)$, $(e_1, \ldots, e_5) = (13, 12, 11, 10, 9)$. As a result,

$$1 + z^2 + z^5 \Bigm| 1 + z^9 + z^{11} + z^{13}.$$

The sequences $(r_1^0, \ldots, r_N^0)$ and $(\Delta_1^0, \ldots, \Delta_N^0)$ presented in Table 3 were constructed under the assumption that $\alpha_1, \ldots, \alpha_{N'}$ are known. The possible algorithm for constructing these vectors repeats the (*D1*)–(*D4*) procedure. This algorithm is given below.

Set $r_1^0 = \ldots = r_N^0 = 0$, $r = \Delta = 0$, $i = j = 1$ and until $j \leq N$ do

($D1'$) if $\alpha_i = +$, then : if $r_j^0 \neq -$, then set $r_j^0 = r$, $\Delta_j^0 = \Delta$, $r = j$, $\Delta = 0$; increase i and j by 1;
($D2'$) if $\alpha_i = D$, then : set $r_j^0 = \ldots = r_{j+m-2}^0 = -$; increase Δ by 1; increase i by 1;
($D3'$) if $\alpha_i = I_b$, where $b \in \{0, 1\}$, then : set $r_j^0 = \ldots = r_{j+m}^0 = -$; decrease Δ by 1; increase i by 1 and j by 2;
($D4'$) if $\alpha_i = S$, then : set $r_j^0 = \ldots = r_{j+m-1}^0 = -$; increase i and j by 1.

The parameter r at each step of the ($D1'$)–($D4'$) algorithm specifies the address of the last investigated row in the matrix consisting of rows $\mathbf{u}_1, \ldots, \mathbf{u}_N$, which contains only non–corrupted symbols of the original sequence $\mathbf{v}|_1^{N'}$, while the pair (r, Δ_r) determines the address of this vector in the matrix consisting of rows $\mathbf{v}_1, \ldots, \mathbf{v}_{N'}$. If $\alpha_i = S$ for the current value of i, then we proceed assuming that $\alpha_{i+1} = \ldots = \alpha_{i+m-1} = +$: we set $r_j^0 = \ldots = r_{j+m-1}^0 = -$, where j is the current index in the matrix consisting of rows $\mathbf{u}_1, \ldots, \mathbf{u}_N$, which corresponds to the index i in the matrix consisting of rows $\mathbf{v}_1, \ldots, \mathbf{v}_{N'}$, since the vectors $\mathbf{u}_j, \ldots, \mathbf{u}_{j+m-1}$ contain the corrupted symbol v_i. If $\alpha_i = D$, then we set $r_i^0 = \ldots = r_{i+m-2}^0 = -$ independently of previous assumptions, because the vectors $\mathbf{u}_j, \ldots, \mathbf{u}_{j+m-1}$ do not contain the

Table 3. Example of the *(D1)*–*(D4)* procedure, $m = 5$, $g(z) = 1 + z^2 + z^5$.

i	α_i	$\mathbf{v}_i$	j	r_j^0, Δ_j^0	$\mathbf{u}_j$	r_j, Δ_j
5	+	00011	5	4,0	00011	4,0
6	+	10001	6	5,0	10001	5,0
7	+	11000	7	6,0	11000	6,0
8	+	11100	8	7,0	11100	7,0
9	+	11110	9	8,0	11110	8,0
10	+	11111	10	9,0	11111	9,0
11	S	01111	11	–	11111	9,−1
12	+	00111	12	–	01111	11,0
13	+	10011	13	–	10111	–
14	D	11001	14	–	01011	–
15	+	01100	15	–	10101	–
16	+	10110	16	–	01010	–
17	+	01011	17	–	00101	–
18	+	00101	18	10,+1	10010	10,+1
19	+	10010	19	18,0	01001	18,0
20	+	01001				

deleted symbol v_i. The cases $\alpha_i = I_0$ and $\alpha_i = I_1$ are similar. The vector $(r_1^0, \ldots, r_N^0)$ constructed by the *(D1′)*–*(D4′)* algorithm has the property that if j^* is the largest index with $r_{j^*} \neq -$, then we can create a chain or "a snake" as follows : while $j^* > 0$ or $r_{j^*} \neq -$, include j^* in the snake and set $j^* = r_{j^*}$.

If $(r_1^0, \ldots, r_N^0)$ is substituted for $(r_1, \ldots, r_N)$ in the algorithm above, then we construct only one snake starting with the largest index j, for which $r_j^0 \neq -$. As a result, all indices j with $r_j^0 \neq -$ will be included in the snake in the decreasing order. However, both the original sequence $\mathbf{v}|_1^{N'}$ and the sequence $(\alpha_1, \ldots, \alpha_{N'})$ are unknown to the receiver, and we cannot assign an algorithm, which requires the check of condition (e) of Lemma 6 at the receiver's side. However, we can run "the divisibility test" where condition (e) is replaced with condition (20). More specifically, suppose that the vectors $(r_1, \ldots, r_{j-1})$ and $(\Delta_1, \ldots, \Delta_{j-1})$ have been constructed by the j–th stage. Then we try to find an integer Δ_j having the smallest absolute value such that there is an $r_j \in \{1, \ldots, N\}$, for which conditions (a)–(d) and (20) are satisfied. If the search is successful for $|\Delta_j| \leq \Delta^*$, where Δ^* is a fixed constant, then r_j and Δ_j are included in the vectors $(r_1, \ldots, r_{j-1})$ and $(\Delta_1, \ldots, \Delta_{j-1})$.

Otherwise, we set $r_j = -$. In both cases j is increased by 1 and the procedure is repeated. Notice that the vector of coefficients $(f_1, \ldots, f_m)$ can be found using methods described in previous section, and the designed algorithm can be effectively implemented. The output vectors $(r_1, \ldots, r_N)$ and $(\Delta_1, \ldots, \Delta_N)$ specify several snakes in general case. We select the longest one and assume that all other m–fragments of the original sequence were corrupted. Additional assumptions concerning possible sequences $(\alpha_1, \ldots, \alpha_{N'})$ can guarantee that one of these snakes always coincides with the snake created by the vectors $(r_1^0, \ldots, r_N^0)$ and $(\Delta_1^0, \ldots, \Delta_N^0)$. In particular, this is true if $|\Delta_j^0| \leq 1$ for all indices j with $r_j^0 \neq -$. Table 3 gives an example of the sequences $(r_5, \ldots, r_{19})$ and $(\Delta_5, \ldots, \Delta_{19})$ constructed under this assumption.

Notice that the situation when neither the feedback polynomial nor the initial content of the shift register is known seems to be more practical. In this case we have to run a procedure where analysis of different feedback polynomials and different initial contents is involved. Our approach can be considered as a tool for solving this problem.

References

1. Rueppel, R. A. (1991) Stream ciphers. In : Contemporary Cryptology (G. J. Simmons, Ed.). New York : IEEE Press. 65–134
2. Massey, J. L. (1969) Shift register synthesis and BCH decoding. IEEE Trans. Inform. Theory. **15**, 122–127
3. Balakirsky, V. B. (1995) On interval linear complexity of binary sequences. SFB–343 "Diskrete Strukturen in der Mathematik", Preprint 97–014. Bielefeld www.mathematik.uni–bielefeld.de/ vbal/
   Proc. 1995 IEEE International Symposium on Information Theory. Whistler, British Columbia, Canada. 490