Cryptanalysis of Nonlinear Feedback Shift Registers

KLAUS POMMERENING

Abstract For a successful cryptanalysis of an NLFSR the needed number of known plaintext bits is about the smaller of two numbers: the period (including the preperiod if the sequence is not purely periodic), and the number of degrees of freedom of the feedback function. Therefore under the assumption that the feedback function is completely unknown there is no better way to cryptanalyse an NLFSR than a straightforward search for the period. If the choice of the feedback function is restricted in order to guarantee its efficient computation, then an algorithm by Boyar and Krawczyk gives an efficient cryptanalysis of the NLFSR in the sense of asymptotic complexity.

Keywords Nonlinear feedback shift register, stream cipher, cryptanalysis, Boyar-Krawczyk algorithm.

Introduction

Feedback Shift Registers (FSR) are useful tools for generating pseudorandom bits [1, 2, 5]. Therefore in cryptography they might be used as key generators for stream ciphers. The main approach to cryptanalyse stream ciphers uses known plaintext and leads to the following problem: Given some bits of the key stream, predict further bits.

The theory of *linear* FSRs is well understood and cryptologically almost trivial [2, 6.2.3]. However for the nonlinear case ((NL)FSR) there are few results in the literature beyond the 1967 monograph by Golomb [1]. One exception is an algorithm by Boyar and Krawczyk (BK) for general congruential generators [4] that also applies to FSRs, i. e. to the case of known modulus 2. The BK algorithm essentially shows in the language of complexity theory: "If a congruential pseudorandom generator is efficiently computable, then it is efficiently predictable."

Surprisingly the following simple result is not found in the open literature: If the cryptanalyst has no information on the feedback function, then

there is no better way for cryptanalysing the FSR than determining the period. Also the BK algorithm, that seeks for linear expressions of unknown bits in terms of known bits, achieves exactly this: It finds the period. A proof is in this paper. From this point of view (NL)FSRs seem cryptographically strong, and even the BK algorithm is not a realistic attack on (NL)FSRs.

From a closer point of view however this strength is illusory. Building an FSR whose configuration parameters allow an arbitrary feedback function is not feasible: There are 2^{2^l} Boolean functions of l Bits—each one is determined by choosing 2^l bits as parameters. Therefore the feedback mechanism must have some restrictions that allow only a restricted ("polynomial in l", the length of the register) choice of the feedback function. By Kerckhoffs' law the cryptanalyst is assumed to know these restrictions. In this situation the BK algorithm provides an asymptotically efficient prediction method. The last part of this paper gives a simple model of "some restrictions" and a simple independent efficiency proof of the BK algorithm for restricted FSRs.

1 Feedback shift registers

We denote by $\mathbb{F}_2$ the field with 2 elements. Mathematically an FSR is modeled as a recursive formula

$$u_n = f(u_{n-1}, \ldots, u_{n-l}) \quad \text{for } n \geq l \tag{1}$$

of depth l that, starting with a vector $(u_0, \ldots, u_{l-1}) \in {\mathbb{F}_2}^l$, generates a bit sequence $u = (u_i)_{i\in\mathbb{N}} \in {\mathbb{F}_2}^{\mathbb{N}}$, see Figure 1. Here $f\colon {\mathbb{F}_2}^l \longrightarrow \mathbb{F}_2$ is a Boolean function of l variables. Its algebraic normal form (ANF) is a polynomial

$$f = \sum_{I\subseteq\{1,\ldots,l\}} a_I T^I \in \mathbb{F}_2[T_1, \ldots, T_l] \quad \text{where } T^I := \prod_{j\in I} T_j\,. \tag{2}$$

We call f the feedback function of the FSR. Note that the number of coefficients a_I is 2^l. Each monomial T^I is given by the corresponding subset $I \subseteq \{1, \ldots, l\}$.

The states of the FSR—i. e. the partial sequences of length l of the output sequence (in reverse order)—constitute the sequence of state vectors

$$u_{(n)} := (u_{n+l-1}, \ldots, u_n) \quad \text{for } n \in \mathbb{N}.$$

By an elementary result on recurrences in finite sets [3, Ex. 3.1.6] the sequence of state vectors has a period ν and a preperiod μ with $0 \leq \mu \leq 2^l - 1$,

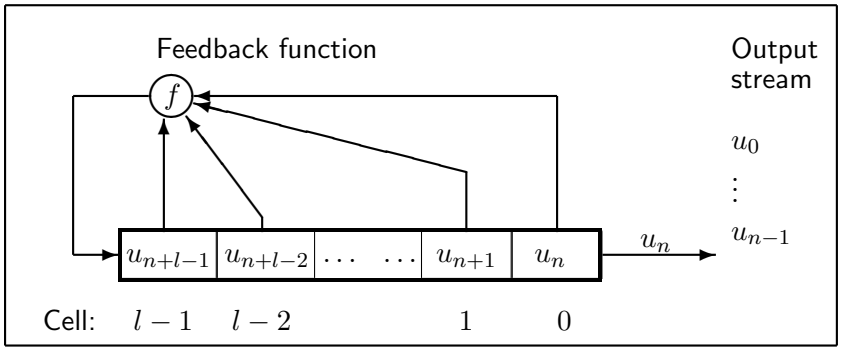


Figure 1: A feedback shift register of length l

$1 \leq \nu \leq 2^l$, $\mu + \nu \leq 2^l$. These numbers also represent the lengths of the preperiod and period of the original FSR sequence u.

The cryptanalyst's goal is to predict the output sequence from some known bits, in particular from a given initial segment. In general she will not succeed in reconstructing the feedback function; however the relevant part of the truth table of the feedback function can be read off from the output sequence.

2 The space of feedback functions

Assume we have a feedback shift register of known length l, but with a completely unknown feedback function.

The set $\mathcal{F}_l$ of all Boolean functions $f\colon {\mathbb{F}_2}^l \longrightarrow \mathbb{F}_2$ of l variables is a vector space over $\mathbb{F}_2$ of dimension 2^l with the monomials T^I, $I \subseteq \{1, \ldots, l\}$ as basis. The simultaneous evaluation map

$$\Psi\colon \mathcal{F}_l \longrightarrow {\mathbb{F}_2}^{2^l}, \quad f \mapsto (f(v))_{v \in {\mathbb{F}_2}^l},$$

is an isomorphism of 2^l-dimensional vector spaces over the field $\mathbb{F}_2$. The image of f is its truth table.

Theorem 1 *The set of feedback functions that generate a given FRS sequence of preperiod μ und period ν in $\mathcal{F}_l$ is an affine subspace of dimension $2^l - \mu - \nu$. In particular there are $2^{2^l-\mu-\nu}$ such functions.*

Proof. Consider the state vectors $u_{(n)}$, $n \in \mathbb{N}$, of the FSR. The given sequence runs through $\mu+\nu$ of them and for each one fixes the value under the feedback

function f by formula (1). Therefore $\mu + \nu$ coordinates of $\Psi(f)$ are fixed. All other coordinates are undetermined. This proves the assertion. $\diamond$

Let us express Theorem 1 in another way: The unknown feedback function has 2^l coefficients. Every subsequence of $l+1$ bits of the output sequence yields one linear equation for these coefficients. Knowing an initial sequence of length $\mu + \nu + l$ gives a system of $\mu + \nu$ linear equations that determines an affine subspace of $\mathcal{F}_l$ of dimension $2^l - \mu - \nu$.

A partial sequence $u_0, \ldots, u_{k-1}$ of an FSR sequence that is not yet periodic runs through $k - l$ distinct state vectors and can, as long as $k - l < 2^l$, be continued in several different ways by filling up the truth table of the feedback function differently. Therefore we have

Corollary 1 *An FSR sequence is uniquely determined by its initial sequence $u_0, \ldots, u_{k-1}$ of length k if and only if $k \geq \mu + \nu + l$.*

If the cryptanalyst knows that many bits, she sees the period, and can correctly predict the whole sequence. There is no method of predicting the sequence that needs fewer than $\mu + \nu + l$ bits. Note that the key length is $2^l + l$—the number of coefficients of the feedback function plus the length of the initial state—not a very useful feature of this cipher, see also Section 9.

For finding the period the cryptanalyst may use the "naive" direct search algorithm or Floyd's algorithm [3, Ex. 3.1.6]. The latter algorithm is somewhat faster and uses less storage but needs even more bits of the sequence. The cost of the direct search per step—i. e. per bit of known plaintext up to the period—consists of comparing the new status vector with all the older ones.

3 FSRs with unknown length

Let's assume the cryptanalyst has a bit sequence $u = (u_0, \ldots, u_{N-1}) \in {\mathbb{F}_2}^N$ and suspects it being generated by an FSR of *unknown* length. Then she successively tries all lengths $l = 1, 2, \ldots$ until she gets a result that fits the entire given sequence. For each l she carries out the following steps:

1. Form the sequence of state vectors $u_{(n)} \in {\mathbb{F}_2}^l$ for $n = 0, \ldots, N - l$ according to the hypothetical FSR length l that is tested.

2. Search for the first repetition in this sequence; this gives values for the preperiod μ and the period ν.

3. Then test the numbers μ und ν whether they give consistent results for the remainder of u.

4. If they are consistent, output l, μ, ν, and stop.

5. Otherwise there is no FSR of length l that generates u. Reject l and try $l + 1$.

This algorithm gives the minimal length l of an FSR that generates the given sequence u together with values for preperiod and period. The sequence of state vectors then gives the truth table of f as completely as is needed to generate u. With these data the cryptanalyst predicts the continuation of the sequence u beyond the index $N - 1$, and uses it to decrypt further portions of the ciphertext. If she later encounters an inconsistency, she has to correct her false prediction by guessing some additional plaintext and then restart the algorithm with the assumed FSR length $l + 1$. For a successful cryptanalysis she needs known plaintext of length $\mu + \nu + l$ bits.

4 The BK algorithm

The algorithm by Boyar and Krawzcyk applies to recursion formulas that have an expression in terms of (unknown) linear combinations of known functions. It considers recursive sequences (s_i) of integers generated by a recursion formula

$$s_i = \sum_{j=1}^{k} c_j \Phi_j(s_0, \ldots, s_{i-1}) \bmod m \quad \text{for } i \geq n_0$$

where

1. the $\Phi_j : \cup_{i \geq n_0} \mathbb{Z}^i \longrightarrow \mathbb{Z}$, $j = 1, \ldots, k$, are known functions,

2. the coefficients c_j are unknown,

3. the modulus m is (known or) unknown.

The algorithm is efficient if the Φ_j are efficiently computable. Its description in the general case is somewhat complicated, so we consider only the case of an FSR of known length l, but unknown feedback function. To this end we consider the feedback function in ANF (2) as a linear combination of the monomials that play the role of the Φ_j. Then the BK algorithm runs as follows:

Embed the state space ${\mathbb{F}_2}^l$ of the FSR nonlinearly into the "extended state space" $Z := {\mathbb{F}_2}^{2^l}$:

$$\Phi\colon \mathbb{F}_2^l \longrightarrow Z, \quad (x_1, \ldots, x_l) \mapsto (x^I)_{I\subseteq\{1,\ldots,l\}}.$$

Enumerate the coordinates in Z canonically from 0 to $2^l - 1$ as follows: The subset $I = \{i_1, \ldots, i_r\} \subseteq \{1, \ldots, l\}$ corresponds to the index

$$j = \sum_{k=1}^{r} 2^{i_k - 1} \quad \text{in binary representation.}$$

In the case $l = 2$ this map is:

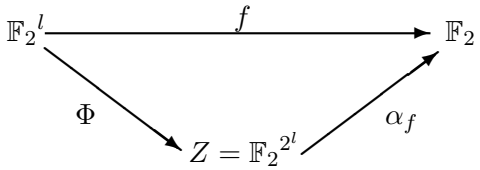
$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ x_1x_2 \end{pmatrix}.$$

Call the image $\Phi(u_{(n)})$ of the state vector $u_{(n)}$ the n-th **extended state vector**.

If $f \in \mathcal{F}_l$ is a Boolean function with ANF (2), then $f(x) = \alpha_f(\Phi(x))$ with the linear map

$$\alpha_f : Z \longrightarrow \mathbb{F}_2, \quad (y_I)_{I\subseteq\{1,\ldots,l\}} \mapsto \sum_{I\subseteq\{1,\ldots,l\}} a_I y_I.$$

(The summation index I runs through the entire power set $\mathfrak{P}(\{1, \ldots, l\})$.) The cryptanalyst doesn't know α_f, that is the 2^l coefficients a_I.



Now let u be an FSR sequence, $z_n = \Phi(u_{(n-l)})$ for all $n \geq l$ be the $(n - l)$-th extended state vector, and $Z_n = \langle z_l, \ldots z_n \rangle$ the vector subspace spanned by these vectors. Then

$$Z_l \subseteq Z_{l+1} \subseteq Z_{l+2} \subseteq \ldots \subseteq Z$$

is an ascending chain of linear subspaces. If $Z_n = Z_{n-1}$, then there is a linear combination

$$z_n = t_l z_l + \cdots + t_{n-1} z_{n-1},$$

that is found by standard linear algebra—solving a system of linear equations in the $\mathbb{F}_2$-vectorspace Z. Because $\alpha_f(z_n) = f(u_{(n-l)}) = u_n$, and α_f is linear, we have

$$u_n = t_l u_l + \cdots + t_{n-1} u_{n-1}.$$

In this way we can predict u_n from the preceding bits of the sequence *without* knowing f. Note that the cost of predicting this single bit consists in solving a system of linear equations in the 2^l-dimensional vector space Z.

Since Z is a finite dimensional vector space, we reach a stationary state after finitely many steps: $Z_k = Z_n =: Z_\infty$ for all $k \geq n$. From this index on all extended state vectors are linear combinations of $z_l, \ldots, z_n$, all following bits of the sequence are uniquely determined and can be predicted from $u_0, \ldots, u_n$. By Corollary 1 this cannot happen before $n = \mu + \nu + l - 1$, i. e. when the period is reached and $\mu + \nu + l$ bits of known plaintext are used. But then it happens indeed, because the extended state vectors have the same periodicity as the simple state vectors. We have shown the first two statements of the following theorem:

Theorem 2 *Let u be an FSR sequence with preperiod μ and period ν. Then:*

(i) [Krawczyk] *If an extended state vector z_n depends linearly from the preceding ones $z_l, \ldots, z_{n-1}$, say $z_n = t_l z_l + \cdots t_{n-1} z_{n-1}$, then $u_n = t_l u_l + t_{n-1} u_{n-1}$.*

(ii) *The ascending chain of subspaces $(Z_n)_{n \geq l}$ spanned by the extended state vectors becomes stationary at $n = \mu + \nu + l - 1$.*

(iii) $\operatorname{Dim} Z_\infty \leq \mu + \nu$.

Proof. Statements (i) and (ii) are already proved. For (iii) note that the increase in dimension in each step can be at most 1, because we use one more vector. $\diamond$

Predicting the next bit is possible whenever the extended state vector depends linearly from the former ones or, in other words, when the sequence of the subspaces Z_n doesn't increase on this step. Note that in the general case of congruential generators considered in [4] each step of the BK algorithm has three possible outcomes:

1. It does't find a linear relation und goes to the next step using one more element of the sequence.

2. It makes a correct prediction.

3. It makes a "mistake", a false prediction. Then the cryptanalyst has to guess one more element of the sequence and to adjust the parameters she calculated before.

The main result of our analysis in the case of modulus 2, i. e. for an FSR of known length, is that the third case cannot occur—we remain in the first case until we find a linear relation and from then on we remain in the second case.

So we ask: Could there be some "early" steps without increase in dimension? In this case the BK algorithm would predict at least some bits correctly *before* it reaches the period. This would be a useful feature for cryptanalysis. However in the next section we prove a theorem that implies the equality in (iii) of Theorem 2. Therefore the BK algorithm finds a linear combination only when the period is reached, and this is then the trivial linear combination $z_{\mu+\nu} = z_\mu$ that expresses the periodicity of the sequence u, and likewise for all vectors to follow.

5 Linear combinations in the BK algorithm

Theorem 3 *The 2^l vectors $\Phi(u)$, $u \in {\mathbb{F}_2}^l$, are a basis of the extended state space $Z = {\mathbb{F}_2}^{2^l}$.*

Proof. Consider the $2^l \times 2^l$ matrix that results from evaluating all 2^l monomials T^I at all 2^l possible argument vectors. The entry in row i and column j is the value of the i-th monomial m_i at the j-th vector x_j for $i, j = 0, \ldots, 2^l - 1$. With the canonical enumeration we have

$$m_i = \prod_{a_k=1} T_k,$$

if $i = a_1 2^{l-1} + \cdots + a_l$ is the representation of the integer i in base 2. Likewise

$$x_j = \begin{pmatrix} b_1 \\ \vdots \\ b_l \end{pmatrix} \quad \text{if } j = b_1 2^{l-1} + \cdots + b_l.$$

Therefore

$$m_i(x_j) = \prod_{a_k=1} b_k = \begin{cases} 1, & \text{if } a_k = 1 \text{ implies } b_k = 1 \text{ for all } k, \\ 0 & \text{else.} \end{cases}$$

This value is 0 when $b_k = 0$ and $a_k = 1$, so always when $j < i$. Therefore the matrix is upper triangular, and all the values in the diagonal $m_i(x_i)$ are equal to 1. Therefore the matrix has rank 2^l, and its columns are a basis of Z. $\diamond$

In particular each set of pairwise distinct extended state vectors in Z is linearly independent, and we conclude:

Corollary 2 *The BK algorithm for an FSR predicts the next bit after exactly $\mu+\nu$ steps, that is, when the period is reached. From then on it predicts all following bits correctly.*

The BK algorithm needs $\mu + \nu + l$ bits, exactly as the naive period calculation does, but it is somewhat slower—in each step it tries to find a linear expression of the new status vector in terms of the older ones instead of simply comparing them. For this reason also the BK algorithm is not useful when considering the case of unknown length l as in Section 3.

6 Restricted feedback functions

All the above results make sense under the assumption that the feedback function can be any Boolean function completely unknown to the cryptanalyst. Defining such a function requires 2^l bits as parameters. If the length is $l \geq 80$ the function is not practically computable, let alone the storage requirements—remember 1 petabyte is only 2^{53} bits. If we want to define an efficiently usable FSR, we have to fix some side conditions that restrict the degrees of freedom in choosing the feedback function. Because these side conditions are part of the algorithm, not part of the key, by Kerckhoffs' law we must assume them known to the enemy. In this setting it also makes little sense to assume the FSR length l as unknown.

As restrictions we consider conditions describing subsets $M \subseteq \mathbb{F}_2[T_1, \ldots, T_l]$ of admissible monomials, and admit feedback functions whose ANFs contain only monomials from M with coefficients 0 or 1. There are 2^m of them where $m = \#M$. Let us call the associated FSR an "M-FSR".

Instead of extended state vectors we now consider "M-state vectors" and the map

$$\Phi_M : {\mathbb{F}_2}^l \longrightarrow {\mathbb{F}_2}^m, \quad u \mapsto (u^I)_{I \in M},$$

that evaluates the vector u at all monomials from M. This map Φ_M consists of Φ followed by the projection π_M from ${\mathbb{F}_2}^{2^l}$ onto the canonically embedded subspace ${\mathbb{F}_2}^m$.

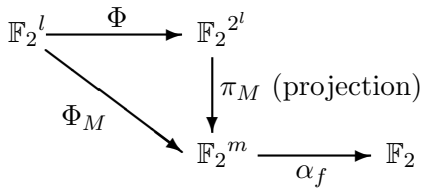
A Boolean function that satisfies our side conditions has the ANF

$$f = \sum_{I \in M} a_I T^I$$

with only m "degrees of freedom". Consider the linear form

$$\alpha_f : {\mathbb{F}_2}^m \longrightarrow \mathbb{F}_2, \quad (y_I)_{I \in M} \mapsto \sum_{I \in M} a_I y_I.$$

Then $f = \alpha_f \circ \Phi_M$.



Now assume we have an FSR sequence $u \in {\mathbb{F}_2}^N$ and the corresponding sequence of state vectors $u_{(0)}, u_{(1)}, u_{(2)}, \ldots$ Then we get a sequence of M-state vectors $z_l, z_{l+1}, z_{l+2}, \ldots$ where $z_k = \Phi_M(u_{(k-l)})$ for $k \geq l$. The BK algorithm looks for a linear relation $z_n = t_l z_l + \cdots t_{n-1} z_{n-1}$ in ${\mathbb{F}_2}^m$; from $\alpha_f(z_n) = f(u_{(n-l)}) = u_n$ we get the relation

$$u_n = t_l u_l + \cdots t_{n-1} u_{n-1},$$

that predicts the bit u_n from the preceding ones at the expense of solving a linear equation in the m-dimensional space of M-state vectors over $\mathbb{F}_2$. This needs about m^3 bit operations when we use Gaussian elimination (and ignore asymptotically faster methods).

We build the ascending chain of subspaces $Z_l \subseteq Z_{l+1} \subseteq \ldots \subseteq \mathbb{F}_2^m$ as in Section 4 and ask at what index n it becomes stationary—this gives the amount of known plaintext that is needed for the attack. We have the same dimension bound $\mu + \nu$ as in Theorem 2, but we also have the bound m because m is the dimension of the space of M-state vectors. Note that—in

contrast to Section 4—the sequence of subspaces is not necessarily strictly increasing, see the example in Section 7. If in an intermediate step we find an equality $Z_i = Z_{i+1}$, then we don't need the next plaintext bit because we can predict it. That means that the amount of needed plaintext is bounded by $\min\{\mu + \nu, m\} + l$. We have shown the following theorem:

Theorem 4 *Let u be an M-FSR sequence with preperiod μ and period ν. Then:*

(i) [Krawczyk] *If an M-state vector z_i depends linearly from the preceding ones $z_l, \ldots, z_{i-1}$, say $z_i = t_l z_l + \cdots t_{i-1} z_{i-1}$, then $u_i = t_l u_l + t_{i-1} u_{i-1}$.*

(ii) *The ascending chain of subspaces $(Z_n)_{n \geq l}$ spanned by the M-state vectors becomes stationary at Z_∞ where* $\operatorname{Dim} Z_\infty \leq \min\{\mu + \nu, m\}$.

The original result by Krawczyk was given in a much more general and abstract setting. In our concrete case we might somewhat sloppily state this result as: The number of bits of known plaintext needed for cryptanalysis is about ($l+$) the number of degrees of freedom of the feedback function. In realistic cases this number is much smaller than the period that is expected to grow exponentially with l.

7 Example

Let us illustrate the effect of Theorem 4 by a toy example. Consider the bit sequence

$$u_0 u_1 u_2 \ldots = 11101\,11101\,11101\ldots \tag{3}$$

that is periodic of period 5. We assume that the cyptanalyst knows

- that this sequence was generated by an FSR of length $l = 4$,
- that at most the monomials from $M = \{1, T_1T_2, T_2T_3, T_3T_4\}$ occur in the ANF (2) of the feedback function,

that is that the feedback function has the form

$$f(x_1, x_2, x_3, x_4) = a_0 + a_{12}x_1x_2 + a_{23}x_2x_3 + a_{34}x_3x_4$$

with unknown coefficients $a_0, a_{12}, a_{23}, a_{34} \in \mathbb{F}_2$. Moreover we assume that she knows a few bits from the beginning of the sequence (3).

The naive attack as in Section 2 tries to find the period. For this the attacker considers the state vectors

$$u_{(0)} = (0,1,1,1), \quad u_{(1)} = (1,0,1,1), \quad u_{(2)} = (1,1,0,1),$$

$$u_{(3)} = (1,1,1,0), \quad u_{(4)} = (1,1,1,1), \quad u_{(5)} = (0,1,1,1),$$

and recognizes the period 5. This attack needs the 9 bits $u_0, \ldots, u_8$. Note that $\mu = 0$, $\nu = 5$, thus $9 = \mu + \nu + l$.

The BK algorithm for M with $m = 4$ uses the map

$$\Phi_M(x_1, x_2, x_3, x_4) = (1, x_1x_2, x_2x_3, x_3x_4).$$

It produces the M-state vectors

$$\begin{aligned} z_4 = \Phi_M(u_{(0)}) &= (1,0,1,1) \\ z_5 = \Phi_M(u_{(1)}) &= (1,0,0,1) \\ z_6 = \Phi_M(u_{(2)}) &= (1,1,0,0) \\ z_7 = \Phi_M(u_{(3)}) &= (1,1,1,0) \\ z_8 = \Phi_M(u_{(4)}) &= (1,1,1,1) \end{aligned}$$

The determinants

$$\det \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \end{pmatrix} = 0, \quad \det \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} = 1$$

show that

- z_7 is in the linear span of z_4, z_5, z_6—in fact $z_7 = z_4 + z_5 + z_6$,
- z_4, z_5, z_6, z_8 constitute a basis of $\mathbb{F}_2^4$.

From z_7 the attacker predicts

$$u_7 = u_4 + u_5 + u_6 = 1.$$

She cannot predict u_8 because z_8 is linearly independent from z_4, z_5, z_6, z_7. However she can predict all further bits u_i, $i \geq 9$. This attack needs the $8 = m + l$ bits $u_0, \ldots, u_6, u_8$.

8 The Asymptotic Complexity

In the example of Section 7 the BK algorithm spared 1 bit of known plaintext at the expense of a lot of Linear Algebra overhead. The usefulness of the BK algorithm in the restricted case becomes evident only for much longer FSRs. This is shown by considering its asymptotic complexity. To do this we consider families of sets of monomials depending on the parameter l.

Definition. A **monomial selection** over $\mathbb{F}_2$ is a family $\mathcal{M} = (M_l)_{l\in\mathbb{N}}$ where each $M_l \subseteq \mathbb{F}_2[T_1, \ldots, T_l]$ is a set of monomials in l variables.

Note. The sets M_l for different l can be completely unrelated in the sense of non-uniform complexity theory.

Definition. $\mathcal{M}$ is called **efficient** when there exists a polynomial $p \in \mathbb{N}[X]$ such that $\#M_l \leq p(l)$ for all $l \in \mathbb{N}$.

Example. Let M_l consist of all monomials up to degree 2. There are $1 + l + \binom{l}{2}$ of them, a number that is clearly polynomial in l.

Note. A more elementary definition of efficiency would use Boolean circuits with l inputs and sizes $\leq$ some polynomials $q(l)$. This would result in an equivalent definition because AND-, OR-, or NOT-gates have simple expressions in terms of the field operations "+" and "·", and vice versa, such as x OR $y = x + y + xy$ for bits $x, y \in \mathbb{F}_2$.

Suppose we have an efficient monomial selection and consider only FSRs whose feedback functions for each length l contain only monomials from M_l. Call these the $\mathcal{M}$-FSRs. Then the cost of the BK algorithm for predicting one bit of an $\mathcal{M}$-FSR of length l is about $p(l)^3$. The preliminary calculations consist of up to $p(l)$ steps, each with up to $p(l)^3$ operations, where prediction is impossible. Hence predicting $q(l)$ bits, where $q \in \mathbb{N}[X]$ is a polynomial, needs at most $p(l)^4 + p(l)^3 q(l)$ calculation steps and up to $p(l) + l$ bits of known plaintext—in other words, the necessary amount of known plaintext has the order of magnitude of the (polynomial) number of degrees of freedom for choosing the feedback function, and the cost is at most a polynomial in the register length. Therefore we can say:

Theorem 5 *The complexity of predicting a polynomially bounded number of bits from $\mathcal{M}$-FSRs is polynomially bounded in the length of the FSR.*

Corollary 3 *FSRs with an efficient monomial selection have an efficient prediction algorithm.*

Or, otherwise expressed, they produce cryptographically weak pseudo-random bits. Note that this result is only a special case of what Krawczyk and Boyar proved, however with a more concrete statement.

Example. A coarse estimate for the example where M consists of the $\approx \frac{1}{2}l^2$ monomials up to degree 2 yields $\approx \frac{1}{16}l^8$ bit operations. For $l = 2^{10} \approx 1000$ this makes $\approx \frac{1}{16} \cdot 2^{80}$—certainly at the limit of of feasibility. We conclude: *The BK algorithm is feasible for registers of lengths up to about 1000 with quadratic feedback functions.*

9 Discussion

Looking at the cryptanalysis of (NL)FSRs we distinguished two cases.

1. The feedback function is completely unknown to the cryptanalyst.
2. The number of variable coefficients of the feedback function is bounded in an efficient way that the cryptanalyst knows.

In the first case the essential criterion for the cryptographic security is the period of the output sequence, for this is the number of algorithmic steps as well as the amount of known plaintext the cryptanalyst needs. This result is in clear contrast to the linear case (LFSR) where $2l$ bits suffice for predicting the whole sequence even if the period is near 2^l (the maximal value for LFSRs is $2^l - 1$ as is well-known [1]). The security of an FSR is given by the parameter $\mu + \nu$, and in "good" cases this has the order of magnitude 2^l, in the average case, the square root of this number, that is $2^{l/2}$ [3, Ex. 3.1.12]. From this viewpoint the attack is realistic for FSRs of length up to 80 in "good" cases, and of length up to 160 in random cases. Therefore in the first case where the costs of an attack grow exponentially with the register length nonlinear feedback shift registers seem reasonably secure in the sense of classical cryptology. But they are not very useful, because of the key size $l + 2^l$; for the critical value $l = 80$ this already takes a key of more than 2^{80} or 10^{24} bits. A realistic plaintext will be much shorter than the period, so the cipher seems perfectly secure. But then why not use a true One Time Pad that needs only as much key bits as the length of the plaintext?

In the second case the BK algorithm is an asymptotically efficient algorithm for cryptanalysis. The amount of needed known plaintext roughly equals the number m of degrees of freedom of the feedback function and the costs grow polynomially with m and the register length l. This might be

prohibitive for a cryptanalysis in a concrete case; however the BK algorithm reveals a significant theoretical weakness in the sense of modern cryptology.

Therefore FSRs should not be directly used as key stream generators for stream ciphers. This recommendation does not preclude the use of FSRs as building blocks for more complicated key stream generators, as a means of introducing some nonlinearity into the design.

One further remark: For the prediction of bitstreams we may also use the Berlekamp Massey (BM) algorithm [2]; it predicts—even without knowing the register length l—all following bits as soon as it reaches the linear complexity λ of the output sequence. This fits the above statements well: The results by Rueppel [5] suggest that the linear complexity in the general case is about 2^{l-1}; we say "suggest" instead of "show", because we consider only FSR sequences, not arbitrary bit sequences. We conjecture that the BM algorithm has no advantage over the BK algorithm in predicting the output of NLFSRs.

About the Author

Klaus Pommerening is a retired professor in Medical Informatics and once implemented some cryptographic infrastructure into german medical research networks. He has taught mathematics at the universities of Heidelberg and Mainz GERMANY after graduating from Freie Universitï¿½t Berlin and receiving his PhD from the University of Mainz.

References

[1] Solomon W. Golomb: *Shift Register Sequences*. Revised Edition: Aegean Park Press 1982.

[2] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone: *Handbook of Applied Cryptography*. CRC Press, Boca Raton 1997.

[3] Donald E. Knuth: *The Art of Computer Programming, Vol II – Seminumerical Algorithms*. Addison-Wesley, Reading 1981.

[4] Hugo Krawczyk: How to predict congruential generators. J. Algorithms 13 (1992), 527–545.

[5] Rainer A. Rueppel: *Analysis and Design of Stream Ciphers*. Springer, Berlin 1986.