# Chapter 3

# LFSR-based Stream Ciphers

In order to minimize the size of the internal state, stream ciphers dedicated to low-cost hardware implementations may use a linear transition function. Among all such possibilities, linear feedback shift registers (LFSRs) offer several advantages including their performance, their implementation cost and many theoretical results on the statistical properties of the produced sequences. LFSR-based generators are then probably the most commonly studied class of keystream generators. This class includes both hardware-oriented stream ciphers and software-oriented ciphers, but this second type of applications usually relies on non-binary LFSRs, operating on a larger alphabet (e.g. on 32-bit words). The most widely used LFSR-based stream ciphers include E0 (used in the Bluetooth standard), A5/1 used for encrypting the over-the-air communications in the GSM cellular telephone standard, SNOW 2.0 (ISO/IEC 18033-4 standard) and its variant SNOW 3G used in UMTS 3G networks.

In most practical LFSR-based generators used nowadays, the internal state is divided into two parts: one is updated linearly by an LFSR, and the other one is updated with a nonlinear function in order to prevent the main attacks exploiting the linearity of the transition function. This nonlinear part may be small (and seen as a nonlinear memory) as in E0 or SNOW 2.0, or both parts of the internal state may be of equal size, like in MUGI or Grain.

## 3.1 Main properties of LFSRs

### 3.1.1 Definitions

An LFSR of length $L$ over $\mathbb{F}_q$ is a finite state automaton which produces a semi-infinite sequence of elements of $\mathbb{F}_q$, $\mathbf{s} = (s_t)_{t\geq 0}$, satisfying a linear recurrence relation of degree $L$ over $\mathbb{F}_q$

$$s_{t+L} = \sum_{i=1}^{L} c_i s_{t+L-i}, \quad \forall t \geq 0 \ .$$

The $L$ coefficients $c_1, \ldots, c_L$ are elements of $\mathbb{F}_q$. They are called the *feedback coefficients* of the LFSR.

The *Fibonacci representation* of an LFSR of length $L$ over $\mathbb{F}_q$ has the form depicted on Figure 3.1. The register consists of $L$ delay cells, called *stages*, each containing an element of $\mathbb{F}_q$. The contents of the $L$ stages, $s_t, \ldots, s_{t+L-1}$, form the *state* of the LFSR. The $L$ stages are initially loaded with $L$ elements, $s_0, \ldots, s_{L-1}$, which can be arbitrary chosen in $\mathbb{F}_q$; they form the initial state of the register.

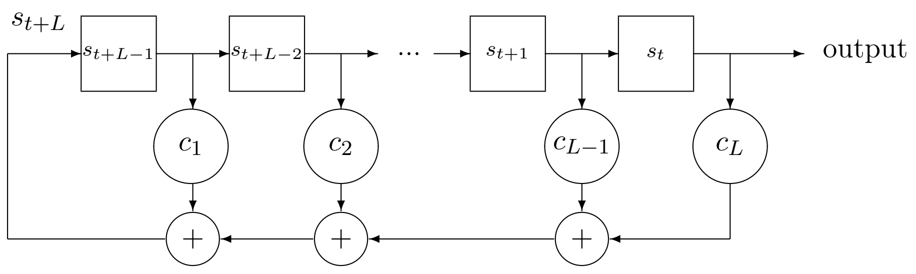


Figure 3.1: Fibonacci representation of an LFSR of length $L$.

The shift register is controlled by an external clock. At each time unit, each digit is shifted one stage to the right. The content of the rightmost stage $s_t$ is output. The new content of the leftmost stage is the *feedback bit*, $s_{t+L}$. It is obtained by a linear combination of the contents of the register stages, where the coefficients of the linear combination are given by the feedback coefficients of the LFSR:

$$s_{t+L} = \sum_{i=1}^{L} c_i s_{t+L-i} \ .$$

Therefore, the LFSR implements the linear recurrence relation of degree $L$:

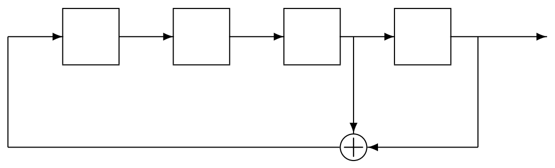
$$s_{t+L} = \sum_{i=1}^{L} c_i s_{t+L-i}, \ \ \forall t \geq 0 \ .$$

**Example 3.1. A binary LFSR of length** 4**.** Table 3.1 gives the successive states of the binary LFSR of length 4 with feedback coefficients $c_1 = c_2 = 0$, $c_3 = c_4 = 1$ and with initial state $(s_0, s_1, s_2, s_3) = (1, 0, 1, 1)$. This LFSR is depicted in Figure 3.2. It corresponds to the linear recurrence relation

$$s_{t+4} = s_{t+1} + s_t \bmod 2 \ .$$

The output sequence $s_0 s_1 \ldots$ generated by this LFSR is $1011100 \ldots$.

Figure 3.2: Binary LFSR with feedback coefficients $(c_1, c_2, c_3, c_4) = (0, 0, 1, 1)$



**Feedback polynomial and characteristic polynomial.** The output sequence of an LFSR is uniquely determined by its feedback coefficients and its initial state. The feedback coefficients $c_1, \ldots, c_L$ of an LFSR of length $L$ are usually represented by the LFSR *feedback polynomial* (or *connection polynomial*) defined by

$$P(X) = 1 - \sum_{i=1}^{L} c_i X^i \ .$$

Table 3.1: Successive states of the LFSR with feedback coefficients $(c_1, c_2, c_3, c_4) = (0, 0, 1, 1)$ and with initial state $(s_0, s_1, s_2, s_3) = (1, 0, 1, 1)$

| $t$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $s_t$ | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| $s_{t+1}$ | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| $s_{t+2}$ | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| $s_{t+3}$ | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |

Alternatively, one can use the *characteristic polynomial*, which is the reciprocal polynomial of the feedback polynomial:

$$P^\star(X) = X^L P(1/X) = X^L - \sum_{i=1}^{L} c_i X^{L-i} \ .$$

For instance, the feedback polynomial of the binary LFSR shown in Figure 3.2 is $P(X) = 1 + X^3 + X^4$ and its characteristic polynomial is $P^\star(X) = 1 + X + X^4$.

**Non-singular LFSRs.** An LFSR is said to be *non-singular* if the degree of its feedback polynomial is equal to the LFSR length (*i.e.*, if the feedback coefficient $c_L$ differs from 0). In this case, the transition function of the LFSR is bijective. Any sequence generated by a non-singular LFSR of length $L$ is periodic, and its period does not exceed $q^L - 1$. Indeed, the LFSR has at most $q^L$ different states and the all-zero state is always followed by the all-zero state. Moreover, if the LFSR is singular, all generated sequences are *ultimately periodic*, *i.e.*, the sequences obtained by ignoring a certain number of elements at the beginning are periodic.

### 3.1.2 Characterization of LFSR output sequences

A given LFSR of length $L$ over $\mathbb{F}_q$ can generate $q^L$ different sequences corresponding to the $q^L$ different initial states and these sequences form a vector space over $\mathbb{F}_q$. The set of all sequences generated by an LFSR with feedback polynomial $P$ is characterized by the following property [Zie59].

**Theorem 3.1.** *A sequence $(s_t)_{t\geq 0}$ is generated by an LFSR of length $L$ over $\mathbb{F}_q$ with feedback polynomial $P$ if and only if there exists a polynomial $Q \in \mathbb{F}_q[X]$ with $\deg(Q) < L$ such that the generating function of $(s_t)_{t\geq 0}$ satisfies*

$$\sum_{t\geq 0} s_t X^t = \frac{Q(X)}{P(X)} \ .$$

*Moreover, the polynomial $Q$ is completely determined by the coefficients of $P$ and by the initial state of the LFSR:*

$$Q(X) = -\sum_{j=0}^{L-1} X^j \left( \sum_{k=0}^{k} s_k c_{j-k} \right) \ ,$$

*where $P(X) = -\sum_{i=0}^{L} c_i X^i$.*

*Proof.*

$$\begin{aligned}\left(-\sum_{i=0}^{L} c_i X^i\right)\left(\sum_{t=0}^{\infty} s_t X^t\right) &= \sum_{j=0}^{\infty} X^j \left(-\sum_{k=\max(0,j-L)}^{j} s_k c_{j-k}\right)\\ &= -\sum_{j=0}^{L-1} X^j \left(\sum_{k=0}^{j} s_k c_{j-k}\right) + \sum_{j=L}^{\infty} X^j \left(\sum_{k=j-L}^{j} s_k c_{j-k}\right) .\end{aligned}$$

Therefore, we deduce that

$$Q(X) = -\left(\sum_{i=0}^{L} c_i X^i\right)\left(\sum_{t=0}^{\infty} s_t X^t\right)$$

is a polynomial of degree strictly less than $L$ if and only if the second right-hand term vanishes, i.e.,

$$\sum_{k=j-L}^{j} s_k c_{j-k} = 0$$

for all $j \geq L$. ◇

This result, which is called the fundamental identity of formal power series of linear recurring sequences, means that there is a one-to-one correspondence between the sequences generated by an LFSR of length $L$ with feedback polynomial $P$ and the fractions $Q(X)/P(X)$ with $\deg(Q) < L$. It has two major consequences. On the first hand, any sequence generated by an LFSR with feedback polynomial $P$ is also generated by any LFSR whose feedback polynomial is a multiple of $P$. This property is widely used for attacking LFSR-based generators (e.g., in distinguishing attacks, and in fast correlation attacks). It may also be helpful since some multiple of the feedback polynomial may provide more appropriate representations in some contexts.

**Example 3.2.** Let $\boldsymbol{s}$ be a binary sequence satisfying

$$s_{t+6} = s_{t+4} + s_{t+3} + s_{t+1} + s_t,\ \forall t \geq 6 .$$

The corresponding feedback polynomial is then $P(X) = 1 + X^2 + X^3 + X^5 + X^6$. It then follows that $\boldsymbol{s}$ also satisfies

$$s_{t+8} = s_{t+7} + s_t$$

since $1 + X + X^8 = (1 + X^2 + X^3 + X^5 + X^6)(1 + X + X^2)$. This alternative recursion may then have a lower implementation cost because of its sparsity.

On the other hand, Theorem 3.1 implies that a sequence generated by an LFSR with feedback polynomial $P$ is also generated by a shorter LFSR with feedback polynomial $P'$ if the corresponding fraction $Q(X)/P(X)$ is such that $\gcd(P, Q) \neq 1$. Thus, amongst all sequences generated by the LFSR with feedback polynomial $P$, there is one which can be generated by a shorter LFSR if and only if $P$ is not irreducible over $\mathbb{F}_q$. This leads to the following natural notion of *minimal polynomial*.

**Definition 3.2.** *For any linear recurring sequence $(s_t)_{t\geq 0}$, there exists a unique polynomial $P_0$ with constant term equal to 1, such that the generating function of $(s_t)_{t\geq 0}$ is given by*

$$\sum_{t\geq 0} s_t X^t = Q_0(X)/P_0(X)$$

*where $P_0$ and $Q_0$ are relatively prime.*

*Then, the shortest LFSR which generates $(s_t)_{t\geq 0}$ has length $L = \max(\deg(P_0), \deg(Q_0)+1)$, and its feedback polynomial is equal to $P_0$. The reciprocal polynomial of $P_0$, $X^L P_0(1/X)$, is the characteristic polynomial of the shortest LFSR which generates $(s_t)_{t\geq 0}$; it is called the* minimal polynomial *of the sequence.*

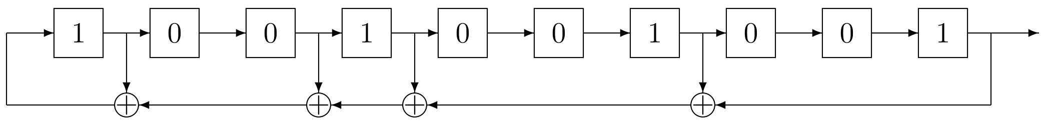
The minimal polynomial of a linear recurring sequence then determines the linear recurrence relation of least degree satisfied by the sequence.

**Example 3.3.** The binary LFSR of length 10 depicted in Figure 3.3 has feedback polynomial

$$P(X) = 1 + X + X^3 + X^4 + X^7 + X^{10} \ ,$$

and its initial state $s_0 \dots s_9$ is 1001001001.

Figure 3.3: Example of a LFSR of length 10.



The generating function of the sequence produced by this LFSR is given by

$$\sum_{t\geq 0} s_t X^t = \frac{Q(X)}{P(X)}$$

where $Q$ is deduced from the coefficients of $P$ and from the initial state:

$$Q(X) = 1 + X + X^7 \ .$$

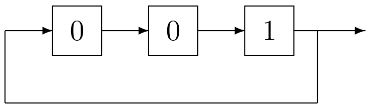
Therefore, we have

$$\sum_{t\geq 0} s_t X^t = \frac{1+X+X^7}{1+X+X^3+X^4+X^7+X^{10}} = \frac{1}{1+X^3} \ ,$$

since $1+X+X^3+X^4+X^7+X^{10} = (1+X+X^7)(1+X^3)$ in $\mathbb{F}_2[X]$. This implies that $(s_t)_{t\geq 0}$ is also generated by the LFSR with feedback polynomial $P_0(X) = 1+X^3$ depicted in Figure 3.4. The minimal polynomial of the sequence is then $1+X^3$.

Obviously, in all cryptographic applications, the feedback polynomials of LFSRs are always chosen irreducible.

Figure 3.4: LFSR of length 3 which generates the same sequence as the LFSR in Figure 3.3



### 3.1.3 Period of a linear recurring sequence

Another important role played by the minimal polynomial is that it determines the period of a linear recurring sequence.

**Proposition 3.3.** *The least period of a linear recurring sequence is equal to the order of its minimal polynomial $P_0$, i.e., to the least positive integer $e$ for which $P_0(X)$ divides $X^e + 1$.*

For instance, the sequence studied in Example 3.3 has minimal polynomial $X^3 + 1$. Then, it has period 3. On the other hand, any non-zero sequence generated by the LFSR of length 4 depicted in Figure 3.2 has period $2^4 - 1 = 15$. Indeed, the minimal polynomial of any such sequence corresponds to its characteristic polynomial $P_0(X) = 1 + X + X^4$, because $P_0$ is irreducible.

We directly deduce from Proposition 3.3 that a sequence has maximal period $2^{\deg P_0} - 1$ if and only if $P_0$ is a *primitive polynomial*. The sequences produced by an LFSR with primitive feedback polynomial are called *maximal-length sequences* (m-sequences).

### 3.1.4 Statistical properties of m-sequences

Maximum-length sequences, i.e., the linear recurring sequences produced by an LFSR with primitive feedback polynomial, possess several good statistical properties which make them appropriate building-blocks in keystream generators.

For instance, any binary sequence produced by an LFSR of length $L$ with primitive feedback polynomial satisfy the following properties. The first three properties are called Golomb's randomness postulates [Gol82].

- *Balance property:* The difference between the number of ones and the number of zeroes in any window of $2^L - 1$ consecutive bits is equal to 1:

$$\#\{i,\ s_{t_0+i} = 1,\ 0 \le i < 2^L - 1\} - \#\{i,\ s_{t_0+i} = 0,\ 0 \le i < 2^L - 1\} = 1\ .$$

- *Runs:* a run is a set of consecutive zeroes flanked by ones, or of consecutive ones flanked by zeroes. For instance, the sequence 0100011 has a run of zeroes of length 3. The proportion of runs of length $i$ within any frame of $(2^L - 1)$ consecutive bits of an m-sequence is $2^{-i}$, $0 \le i < L$. Moreover, among all runs of length $i$, $i \le L - 2$, the number of runs of zeroes and the number of runs of ones are equal. There is exactly one run of length $(L - 1)$ and one run of length $L$ (see Example 3.4).

- *Two-level auto-correlation:* The autocorrelation of a binary sequence of period $N$ is defined by

$$C(\tau) = \sum_{t=0}^{2^L-2} (-1)^{s_t + s_{t+\tau \bmod (2^L-1)}}\ .$$

$C(\tau)$ then measures the distance between the sequence $s_0s_1 \dots s_{2^L-2}$ and the sequence obtained by shifting it by $\tau$ positions, i.e., $s_\tau s_{\tau+1} \dots s_{\tau-1}$. Indeed, we have

$$C(\tau) = 2^L - 1 - 2\#\{t,\ s_t \neq s_{t+\tau \bmod (2^L-1)}, 0 \leq t < 2^L - 1\} .$$

In particular, $C(\tau) = 2^L - 1$ if and only if $\tau$ is a multiple of the period of the sequence since the two sequences are identical. For any m-sequence generated by an LFSR with primitive feedback polynomial of degree $L$, we have that, if $\tau$ is not a multiple of $(2^L-1)$, then $C(\tau) = -1$. This means that the sequence is as far as possible from its shifted versions. This property is widely used in telecommunications for synchronization.

- *Multigram property:* the $L$-tuple $s_ts_{t+1} \dots s_{t+L-1}$ takes all the $2^L - 1$ nonzero values when $t$ varies between 0 and $(2^L - 2)$.

Some additional properties of m-sequences are detailed in [Hel11].

**Example 3.4.** Let us consider the first 31 bits of the binary sequence produced by the LFSR of length 5 with primitive feedback polynomial $X^5 + X^3 + 1$ from initial state 10000:

$$1000010101110110001111100110100$$

It can be checked that this sequence consists of 16 ones and 15 zeroes. It then satisfies the balance property.

The sequence has 16 runs: 8 runs of length 1 (4 runs of zeroes and 4 runs of ones), 4 runs of length 2 (2 runs of zeroes and 2 runs of ones), 2 runs of length 3 (1 run of zeroes and 1 run of ones), one run of zeroes of length 4 and one run of ones of length 5.

### 3.1.5 LFSR and multiplication in a finite field

The operation performed by a $q$-ary LFSR of length $L$ with irreducible feedback polynomial corresponds to a multiplication in the finite field $\mathbb{F}_{q^L}$.

**Proposition 3.4.** *Let $P^\star$ be an irreducible polynomial in $\mathbb{F}_q[X]$ with degree $L$. Let $\alpha \in \mathbb{F}_{q^L}$ be a root of $P^\star$ and $\{\beta_0, \dots, \beta_{L-1}\}$ denote the dual basis of $\{1, \alpha, \dots, \alpha^{L-1}\}$, i.e.,*

$$\mathsf{Tr}(\alpha^i\beta_j) = \begin{cases} 0 & \text{if } i \neq i \\ 1 & \text{if } i = j \end{cases} ,$$

*where $\mathsf{Tr}$ denotes the trace function from $\mathbb{F}_{q^L}$ into $\mathbb{F}_q$.*

*Then, the content of the LFSR with characteristic polynomial $P^\star$ at time $(t+1)$ is equal to its content at time $t$ multiplied by $\alpha$, where these vectors are identified with elements in the field $\mathbb{F}_{q^L}$ decomposed on the basis $\{\beta_0, \dots, \beta_{L-1}\}$.*

*Proof.* For any $t$, we identify an $L$-tuple $(x_0, \dots, x_{L-1})$ with an element in the finite field $\mathbb{F}_{q^L}$ by

$$x = x_{L-1}\beta_{n-1} + \dots + x_0\beta_0 .$$

Then, by definition of the dual basis, we have that, for any $0 \leq i < L$,

$$\begin{aligned} \mathsf{Tr}(\alpha^i x) &= \sum_{j=0}^{L-1} \mathsf{Tr}(x_j\alpha^i\beta_j) \\ &= \sum_{j=0}^{L-1} x_j \mathsf{Tr}(\alpha^i\beta_j) = x_i . \end{aligned}$$

This means that the $i$-th coordinate of $x$ in the basis $\{\beta_0, \ldots, \beta_{L-1}\}$ is equal to $\mathsf{Tr}(\alpha^i x)$. Let us now compute the coordinates of $y = \alpha x$ in this basis. The $i$-th coordinate of $y$ is given by

$$\mathsf{Tr}(\alpha^i y) = \mathsf{Tr}(\alpha^{i+1} x) \ .$$

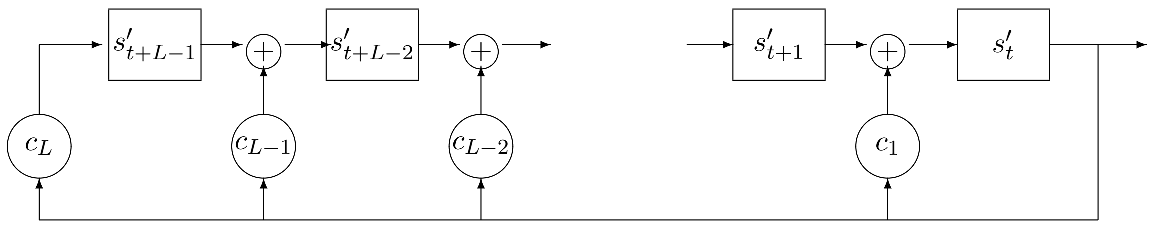
It follows that the $i$-th coordinate of $y$ is equal to the $(i+1)$-th coordinate of $x$ if $i < L-1$. For $i = L-1$, the last coordinate of $y$ is given by

$$\begin{aligned} \mathsf{Tr}(\alpha^{L-1} y) &= \mathsf{Tr}(\alpha^L x) \\ &= \mathsf{Tr}\left(\sum_{i=1}^{L} c_i \alpha^{L-i} x\right) \\ &= \sum_{i=1}^{L} c_i \mathsf{Tr}(\alpha^{L-i} x) = \sum_{i=1}^{L} c_i x_{L-i} \ , \end{aligned}$$

where the characteristic polynomial is given by $P^\star(X) = X^L - \sum_{i=1}^{L} c_i X^{L-i}$, implying that $\alpha^L = \sum_{i=1}^{L} c_i \alpha^{L-i}$. It follows that the coordinates of $y = \alpha x$ correspond to the content after one clock of the LFSR initialized by $x$. $\diamond$

**Galois representation.** Since an LFSR with irreducible feedback polynomial is a device which implements the multiplication by an element $\alpha$ in a finite field, some alternate automorphism between the $\mathbb{F}_{q^L}$ and $\mathbb{F}_q^L$ may be used without modifying the transition function over $\mathbb{F}_{q^L}$. Another natural representation is obtained when the basis $\{1, \alpha, \alpha^2, \ldots, \alpha^{L-1}\}$ is used for representing the elements in $\mathbb{F}_{q^L}$ instead of the dual basis $\{\beta_0, \ldots, \beta_{L-1}\}$. This representation is called the *Galois representation*, and corresponds to the "natural" multiplication circuit, i.e., the multiplication in the so-called polynomial basis. The Galois representation corresponding to the Fibonacci LFSR depicted on Figure 3.1 is given in Figure 3.5.

Figure 3.5: Galois representation of the LFSR depicted on Figure 3.1.



It is worth noticing that Fibonacci and Galois representations have different features. For instance, the Galois representation is obviously more efficient in software than the Fibonacci representation. Also, the diffusion within the register in the Galois representation is faster.

## 3.2 Linear complexity and LFSR synthesis

A fundamental quantity for a sequence is its linear complexity since it determines the smallest linear recursion satisfied by the sequence, or equivalently the length of the smallest LFSR generating the sequence.

**Definition 3.5.** *The* linear complexity *of a semi-infinite sequence* $\mathbf{s} = (s_t)_{t \geq 0}$ *of elements of* $\mathbb{F}_q$, $\Lambda(\mathbf{s})$, *is the smallest integer* $\Lambda$ *such that* $\mathbf{s}$ *can be generated by an LFSR of length* $\Lambda$ *over* $\mathbb{F}_q$, *and is* $\infty$ *if no such LFSR exists. By way of convention, the linear complexity of the all-zero sequence is equal to* $0$. *The linear complexity of a linear recurring sequence corresponds to the degree of its minimal polynomial.*

*The linear complexity* $\Lambda(\mathbf{s}^n)$ *of a finite sequence* $\mathbf{s}^n = s_0 s_1 \ldots s_{n-1}$ *of* $n$ *elements of* $\mathbf{F}_q$ *is the length of the shortest LFSR which produces* $\mathbf{s}^n$ *as its first* $n$ *output terms for some initial state.*

The linear complexity of an infinite linear recurring sequence $\mathbf{s}$ and the linear complexity of the finite sequence $\mathbf{s}^n$ composed of the first $n$ digits of $\mathbf{s}$ are related by the following property: if $\mathbf{s}$ is an infinite linear recurring sequence with linear complexity $\Lambda$, then the finite sequence $\mathbf{s}^n$ has linear complexity $\Lambda$ for any $n \geq 2\Lambda$ [Mas69]. Moreover, the unique LFSR of length $\Lambda$ that generates $\mathbf{s}$ is the unique LFSR of length $\Lambda$ that generates $\mathbf{s}^n$ for every $n \geq 2\Lambda$.

### 3.2.1 Linear complexity as a statistical test

For a sequence $\mathbf{s} = s_0 s_1 \ldots$, the sequence of the linear complexities $(\Lambda(\mathbf{s^n}))_{n \geq 1}$ of all subsequences $\mathbf{s}^n = s_0 \ldots s_{n-1}$ composed of the first $n$ terms of $\mathbf{s}$ is called the *linear complexity profile* of $\mathbf{s}$.

**Proposition 3.6.** *[Rue86, Page 40] The expected linear complexity of a binary sequence* $\mathbf{s}^n = s_0 \ldots s_{n-1}$ *of* $n$ *independent and uniformly distributed binary random variables is*

$$E[\Lambda(\mathbf{s}^n)] = \frac{n}{2} + \frac{4 + \varepsilon(n)}{18} + 2^{-n}\left(\frac{n}{3} + \frac{2}{9}\right) ,$$

*where* $\varepsilon(n) = n \bmod 2$.

Therefore, it may be possible to distinguish a sequence from a truly random sequence by computing its linear complexity profile and comparing the result to what is expected from Proposition 3.6. Further results on the linear complexity and on the linear complexity profile of random sequences can be found in [Rue86].

### 3.2.2 Berlekamp-Massey algorithm

The *Berlekamp-Massey algorithm* is an algorithm for determining the linear complexity of a finite sequence and the feedback polynomial of an LFSR of minimal length which generates this sequence. This algorithm is due to Massey [Mas69], who showed that the iterative algorithm proposed in 1967 by Berlekamp [Ber68] for decoding BCH codes can be used for finding the shortest LFSR that generates a given sequence. Given sequence $\mathbf{s}^n$ of length $n$, the Berlekamp-Massey performs $n$ iterations. The $t$-th iteration determines an LFSR of minimal length which generates the first $t$ digits of $\mathbf{s}^n$. The algorithm is described in Algorithm 7. In the particular case of a binary sequence, the quantity $d'$ does not need to be stored since it is always equal to 1. Moreover, the feedback polynomial is simply updated by

$$P(X) \leftarrow P(X) + P'(X)X^{t-m} .$$

The number of operations performed for computing the linear complexity of a sequence of length $n$ is $\mathcal{O}(n^2)$. It can be proved that the Berlekamp-Massey algorithm and the Euclidean algorithm are essentially the same [Dor87].

**Algorithm 7** The Berlekamp-Massey algorithm.

```
Input: $\mathbf{s}^n = s_0 s_1 \dots s_{n-1}$, a sequence of $n$ elements of $\mathbb{F}_q$.
Output: $\Lambda$, the linear complexity of $\mathbf{s^n}$ and $P$, the feedback polynomial of an LFSR of
length $\Lambda$ which generates $\mathbf{s}^n$.
/* Initialization */
$P(X) \leftarrow 1$, $P'(X) \leftarrow 1$, $\Lambda \leftarrow 0$, $m \leftarrow -1$, $d' \leftarrow 1$.
/* Algorithm */
for $t$ from 0 to $n-1$ do
  $d \leftarrow s_t + \sum_{i=1}^{\Lambda} p_i s_{t-i}$.
  if $d \neq 0$ then
    $T(X) \leftarrow P(X)$.
    $P(X) \leftarrow P(X) - d(d')^{-1} P'(X) X^{t-m}$.
    if $2\Lambda \leq t$ then
      $\Lambda \leftarrow t + 1 - \Lambda$.
      $m \leftarrow t$.
      $P'(X) \leftarrow T(X)$.
      $d' \leftarrow d$.
    end if
  end if
end for
return $\Lambda$ and $P$
```

The LFSR of minimal length that generates a sequence $\mathbf{s}^n$ of length $n$ is unique if and only if $n \geq 2\Lambda(\mathbf{s}^n)$, where $\Lambda(\mathbf{s}^n)$ is the linear complexity of $\mathbf{s}^n$.

Obviously, the linear complexity $\Lambda(\mathbf{s})$ of a semi-infinite linear recurring sequence $\mathbf{s} = (s_t)_{t \geq 0}$ is equal to the linear complexity of the finite sequence composed of the first $n$ terms of $\mathbf{s}$ for any $n \geq \Lambda(\mathbf{s})$. Thus, the Berlekamp-Massey algorithm determines the shortest LFSR that generates an infinite linear recurring sequence $\mathbf{s}$ from the knowledge of any $2\Lambda(\mathbf{s})$ consecutive digits of $\mathbf{s}$.

**Example 3.5.** Table 3.2 describes the successive steps of the Berlekamp-Massey algorithm applied to the binary sequence of length 7, $s_0 \dots s_6 = 0111100$. The values of $\Lambda$ and $P$

| $t$ | $s_t$ | $d$ | $\Lambda$ | $P(X)$ | $m$ | $P'(X)$ |
|---|---|---|---|---|---|---|
|  |  |  | 0 | 1 | $-1$ | 1 |
| 0 | 0 | 0 | 0 | 1 | $-1$ | 1 |
| 1 | 1 | 1 | 2 | $1 + X^2$ | 1 | 1 |
| 2 | 1 | 1 | 2 | $1 + X + X^2$ | 1 | 1 |
| 3 | 1 | 1 | 2 | $1 + X$ | 1 | 1 |
| 4 | 1 | 0 | 2 | $1 + X$ | 1 | 1 |
| 5 | 0 | 1 | 4 | $1 + X + X^4$ | 5 | $1 + X$ |
| 6 | 0 | 0 | 4 | $1 + X + X^4$ | 5 | $1 + X$ |

Table 3.2: Successive steps of the Berlekamp-Massey algorithm applied to the binary sequence of length 7, $s_0 \dots s_6 = 0111100$.

obtained at the end of step $t$ correspond to the linear complexity of the sequence $s_0 \dots s_t$ and to the feedback polynomial of an LFSR of minimal length that generates it.

## 3.3 Classical constructions of LFSR-based generators

It is clear that an LFSR should never be used by itself as a keystream generator. If the feedback coefficients are publicly known (which is usually the case when the LFSR is implemented in hardware), the entire keystream can obviously be recovered from the knowledge of any $\Lambda$ consecutive bits of the keystream, where $\Lambda$ is the linear complexity of the running-key (which does not exceed the LFSR length). If the feedback coefficients are kept secret, the entire keystream can be recovered from any $2\Lambda$ consecutive bits of the keystream by the Berlekamp-Massey algorithm.

However, LFSRs are extremely fast and low-cost devices and they generate sequences with good statistical properties, in particular with a high period. Therefore, they are often used as building-blocks in dedicated keystream generators, but within a more complex system. In particular, three classical constructions based on LFSR aim at increasing the linear complexity of the generated sequence at a low implementation cost. These three methods have received a lot of attention and have been widely used within stream ciphers. They include the combination generator, the filter generator and the generators based on LFSR with irregular clocking.

### 3.3.1 Combination generators

A *combination generator* is a keystream generator composed of several LFSRs whose outputs are combined by a Boolean function to produce the keystream. Then, the output sequence $(s_t)_{t\geq 0}$ of a combination generator composed of $n$ LFSRs is given by

$$s_t = f(u_t^1, u_t^2, \dots, u_t^n), \ \ \forall t \geq 0 \ ,$$

where $(u_t^i)_{t\geq 0}$ denotes the sequence generated by the $i$-th constituent LFSR and $f$ is a function of $n$ variables. In the case of a combination generator composed of $n$ LFSR over $\mathbb{F}_q$, the combining function is a function from $\mathbf{F}_q^n$ into $\mathbb{F}_q$.

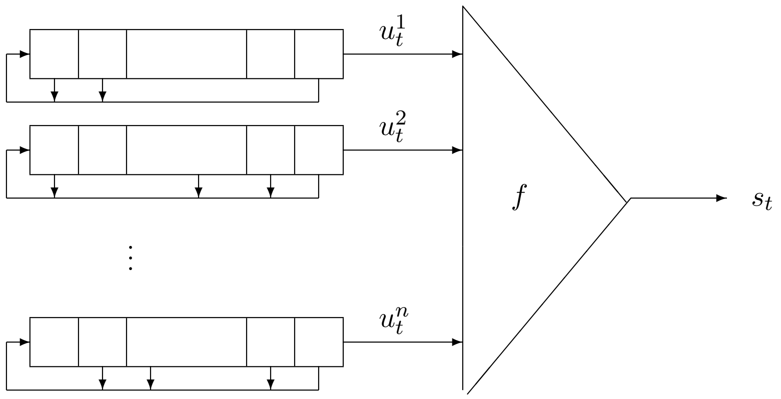


Figure 3.6: Combination generator.

The combining function $f$ should obviously be *balanced*, i.e., its output should be uniformly distributed. The constituent LFSRs should be chosen to have primitive feedback polynomials for ensuring good statistical properties of their output sequences. The characteristics of the constituent LFSRs and the combining function are usually publicly known. The secret parameters are the initial states of the LFSRs, which are derived from the secret key of the cipher by a key-loading algorithm. When the feedback polynomials of the LFSR and the combining function are not known, the reconstruction attack presented in [CF01] enables to recover the complete description of the generator from the knowledge of a large segment of the ciphertext sequence.

**Example 3.6. Geffe generator [Gef73]** The generator proposed by Geffe [Gef73] is composed of three LFSRs of distinct lengths combined by the function

$$f(x_1, x_2, x_3) = x_1x_2 + x_2x_3 + x_3 \ .$$

It is worth noticing that this function corresponds to an IF: if $x_2 = 0$, then $f(x_1, x_2, x_3) = x_3$ and if $x_2 = 1$, $f(x_1, x_2, x_3) = x_1$.

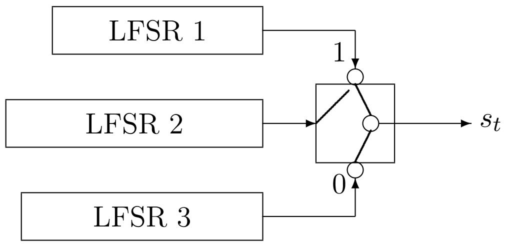


Figure 3.7: Geffe generator.

**Linear complexity of the output sequence.** The sequence produced by a combination generator is a linear recurring sequence. Its period and its linear complexity can be derived from those of the sequences generated by the constituent LFSRs and from the ANF of the combining function. Indeed, if we consider two linear recurring sequences $\mathbf{u}$ and $\mathbf{v}$ over $\mathbb{F}_q$ with linear complexities $\Lambda(\mathbf{u})$ and $\Lambda(\mathbf{v})$, we have the following properties:

- the linear complexity of the sequence $\mathbf{u} + \mathbf{v} = (u_t + v_t)_{t\geq 0}$ satisfies

  $$\Lambda(\mathbf{u} + \mathbf{v}) \leq \Lambda(\mathbf{u}) + \Lambda(\mathbf{v}) \ ,$$

  with equality if and only if the minimal polynomials of $\mathbf{u}$ and $\mathbf{v}$ are relatively prime. Moreover, in case of equality, the period of $\mathbf{u} + \mathbf{v}$ is the least common multiple of the periods of $\mathbf{u}$ and $\mathbf{v}$.

- the linear complexity of the sequence $\mathbf{uv} = (u_tv_t)_{t\geq 0}$ satisfies

  $$\Lambda(\mathbf{uv}) \leq \Lambda(\mathbf{u})\Lambda(\mathbf{v}) \ ,$$

  where equality holds if the minimal polynomials of $\mathbf{u}$ and $\mathbf{v}$ are primitive and if $\Lambda(\mathbf{u})$ and $\Lambda(\mathbf{v})$ are distinct and greater than 2. Other general sufficient conditions for $\Lambda(\mathbf{uv}) = \Lambda(\mathbf{u})\Lambda(\mathbf{v})$ can be found in [Her86, RS87, GN95].

These results lead to the following general proposition.

**Proposition 3.7.** *[RS87] Let us consider the combination generator composed of $n$ binary LFSRs with primitive feedback polynomials which are combined by a Boolean function $f$. If all LFSR lengths $L_1, \ldots, L_n$ are distinct and greater than 2 (and if the LFSR initializations differ from the all-zero state), the linear complexity of the output sequence* $\mathbf{s}$ *is equal to*

$$f(L_1, L_2, \ldots, L_n)$$

*where the algebraic normal form of $f$ is evaluated over integers.*

For instance, if four LFSRs of lengths $L_1, \ldots, L_4$ satisfying the previous conditions are combined by the Boolean function $x_1x_2 + x_2x_3 + x_4$, the linear complexity of the resulting sequence is $L_1L_2 + L_2L_3 + L_4$. Similar results concerning the combination of LFSRs over $\mathbb{F}_q$ can be found in [Bry86, GN95]. For instance, the linear complexity of the sequence produced by the Geffe generator is $L_1L_2 + L_2L_3 + L_3$ where $L_1$, $L_2$ and $L_3$ denote the lengths of the constituent LFSRs.

A high linear complexity is obviously desirable for a keystream sequence since it ensures that Berlekamp-Massey algorithm becomes computationally infeasible. Thus, the combining function $f$ should have a high *algebraic degree.*

A detailed analysis of the security of the combination generator, especially its resistance to correlation attacks, is provided in Section 3.5.

### 3.3.2 Filter generator

A *filter generator* is a keystream generator composed of a single LFSR whose content is filtered by a nonlinear function. More precisely, the output sequence of a filter generator corresponds to the output of a nonlinear function whose inputs are taken from some stages of the LFSR. If $(u_t)_{t\geq 0}$ denotes the sequence generated by the LFSR, the output sequence $(s_t)_{t\geq 0}$ of the filter generator is given by

$$s_t = f(u_{t+\gamma_1}, u_{t+\gamma_2}, \ldots, u_{t+\gamma_n}), \ \ \forall t \geq 0$$

where $f$ is a function of $n$ variables, $n$ is less than or equal to the LFSR length and $(\gamma_i)_{1\leq i\leq n}$ is a decreasing sequence of non-negative integers called the *tapping sequence.*

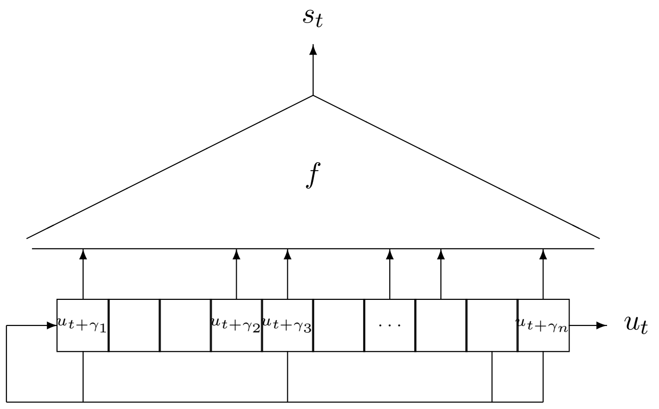


Figure 3.8: Filter generator.

In order to obtain a keystream sequence having good statistical properties, the filtering function $f$ should be *balanced*, and the feedback polynomial of the LFSR should be chosen to be a primitive polynomial.

In a filter generator, the LFSR feedback polynomial, the filtering function and the tapping sequence are usually publicly known. The secret parameter is the initial state of the LFSR which is derived from the secret key of the cipher. The attack presented in [Sie85] enables to construct an equivalent keystream generator from the knowledge of a large segment of the ciphertext sequence when the LFSR feedback polynomial is the only known parameter (i.e., when the filtering function, the tapping sequence and the initial state are kept secret).

Any filter generator is equivalent to a particular combination generator, in the sense that both generators produce the same output sequence: an equivalent combination generator consists of $n$ copies of the LFSR used in the filter generator with shifted initial states; the combining function corresponds to the filtering function.

**Linear complexity of the output sequence.** The output sequence $\mathbf{s}$ of a filter generator is a linear recurring sequence. Its linear complexity, $\Lambda(\mathbf{s})$, is related to the LFSR length and to the algebraic degree of the filtering function $f$. For a binary LFSR with a primitive feedback polynomial, we have

$$\Lambda(\mathbf{s}) \leq \sum_{i=0}^{d} \binom{L}{i}$$

where $L$ denotes the LFSR length and $d$ denotes the algebraic degree of $f$ [Key76, Mas01]. The period of $\mathbf{s}$ divides $2^L - 1$. Moreover, if $L$ is a large prime, $\Lambda(\mathbf{s})$ is at least $\binom{L}{d}$ for most filtering functions with algebraic degree $d$ [Rue86]. To achieve a high linear complexity, the LFSR length $L$ and the algebraic degree of the filtering function should be large enough. More precisely, the keystream length available to an attacker should always be much smaller than $\binom{L}{\deg(f)}$.

It is worth noticing that the lower bound on the linear complexity does not hold for all functions. For instance, some examples of filter generators with an LFSR of size $L$ and a filtering function of high degree, but with linear complexity $L$ only can be exhibited [RC10].

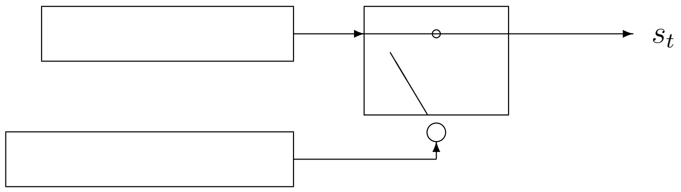
### 3.3.3 LFSRs with irregular clocking

Another technique for increasing the linear complexity of the produced keystream consists in considering one or several LFSRs, but some LFSR bits decide which LFSR to clock and how often. The most prominent example is the shrinking generator proposed in 1993 by Coppersmith, Krawczyk and Mansour [CKM94]. It is composed of two LFSRs, and the output of the second LFSR controls the clock of the first one. More precisely, if the second LFSR outputs 0, the output bit of the first one is discarded. This generator is depicted on Figure 3.9. Then, it can be proved that the linear complexity of the produced sequence is at least

$$L_A 2^{L_B - 2}$$

where $L_A$ and $L_B$ denote the linear complexities of the two constituent registers. The self-shrinking generator [MS95] and the alternating-step generator [Gün88] are two other examples of clock-controlled generators.

Figure 3.9: The shrinking generator.



## 3.4 Some widely-used LFSR-based generators

### 3.4.1 A5/1

*A5/1* is the stream cipher used for encrypting over-the-air transmissions in the GSM standard. A5/1 is used in most European countries, whereas a weaker cipher, called A5/2, is used in other countries (a description of A5/2 and an attack can be found in [PFS00]). The description of A5/1 was first kept secret but its design has been reverse-engineered in 1999 by Briceno, Golberg and Wagner [BGW99].

A5/1 has a 64-bit secret key. A GSM conversation is transmitted as a sequence of 228-bit frames (114 bits in each direction) every 4.6 millisecond. Each frame is xored with a 228-bit sequence produced by the A5/1 running-key generator. The initial state of this generator depends on the 64-bit secret key, $K$, which is fixed during the conversation, and on a 22-bit public *frame number*, $F$.

**Description of the running-key generator.** The A5/1 running-key generator is composed of 3 LFSRs of lengths 19, 22 and 23. Their characteristic polynomials are $X^{19}+X^5+X^2+X+1$, $X^{22}+X+1$ and $X^{23}+X^{15}+X^2+X+1$. The internal state of the generator then consists of 64 bits only, which makes it vulnerable to Time-Memory-Data-Trade-off attacks.

For each frame transmission, the 3 LFSRs are first initialized to zero. Then, at time $t=1,\dots,64$, the LFSRs are clocked, and the key bit $K_t$ is xored to the feedback bit of each LFSR. For $t=65,\dots,86$, the LFSRs are clocked in the same fashion, but the $(t-64)$-th bit of the frame number is now xored to the feedback bits. This initialization phase is depicted on Figure 3.10.

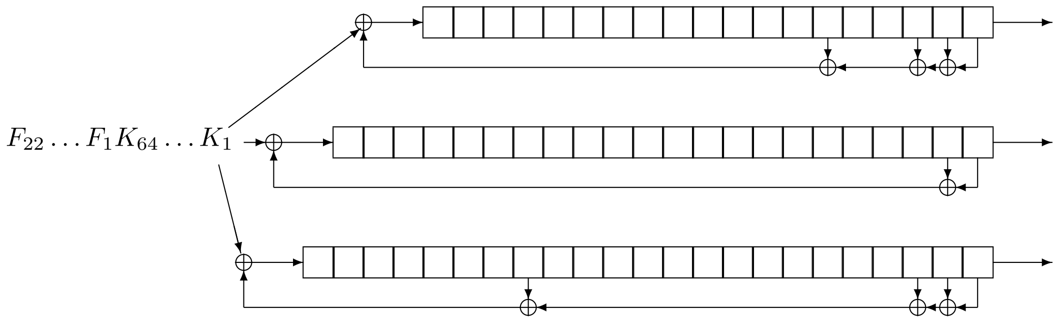


Figure 3.10: Initialization of the A5/1 running-key generator.

After these 86 cycles, the generator runs as depicted on Figure 3.11. Each LFSR has a clocking tap: tap 8 for the first LFSR, tap 10 for the second and the third ones (where the feedback tap corresponds to tap 0). At each unit of time, the majority value $b$ of the 3 clocking bits is computed. A LFSR is clocked if and only if its clocking bit is equal to $b$. For instance, if the 3 clocking bits are equal to $(1, 0, 0)$, the majority value is 0. The second and third LFSRs are clocked, but not the first one. The output of the generator is then given by the xor of the outputs of the 3 LFSRs. After the 86 initialization cycles, 328 bits are generated with the previously described irregular clocking. The first 100 ones are discarded and the following 228 bits form the running-key.

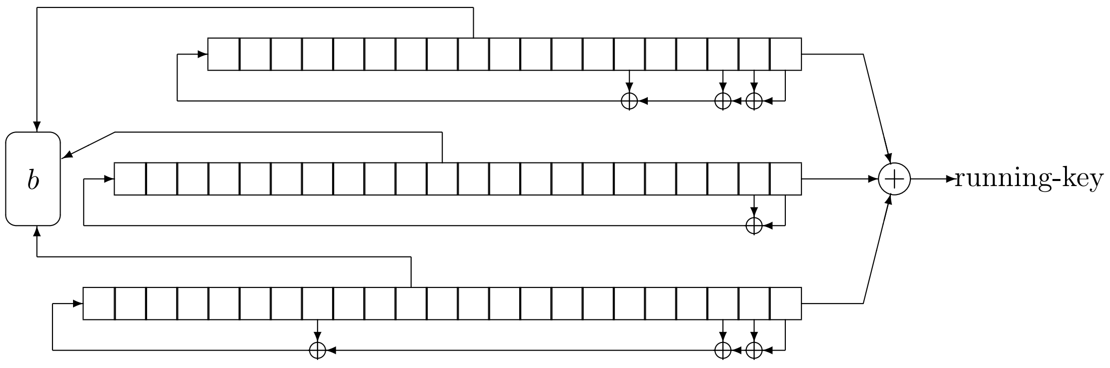


Figure 3.11: A5/1 running-key generator.

**Attacks on A5/1.** Several time-memory trade-off attacks have been proposed on A5/1 exploiting the small size of the secret key or of the internal state [BD00, BSW00]. They require the knowledge of a few seconds of conversation plaintext and run very fast. Even if they need a huge precomputation time and memory, an optimized version has been implemented in 2008: the group *The Hacker's Choice* has precomputed the huge look-up tables involved in the time-memory-trade-off attack. These tables have also been computed and then released in December 2009 by the *A5/1 cracking project* [NP09], and an improved implementation has been described in [KPPM12].

Another attack due to Ekdahl and Johansson [EJ03] exploits some weaknesses of the key initialization procedure. It has been later improved by Maximov, Johansson and Babbage [MJB04] and then by Barkan and Biham [BB06]. It requires a few minutes using 5-10 seconds of conversation plaintext without any notable precomputation and storage capacity. Most of these attacks can also be turned into ciphertext-only attacks in the context of GSM communications by exploiting the fact that error-correction is performed before encryption in the GSM transmissions [BBK08].

### 3.4.2 E0

E0 is the stream cipher used for ensuring the confidentiality of communications in the Bluetooth protocol for wireless short-range connectivity [Blu].

The keysize in E0 is 128 bits. More precisely, the number of key bytes, between 1 and 16 is negotiated between the two modules in the protocol, but the key is always extended to a 128-bit word by adding some redundancy when the effective number of key bits is less than

128. The IV has 64 bits which correspond to the 48-bit Bluetooth address, and a 26-bit master counter.

In the Bluetooth protocol, data are transmitted as frames of at most 2745 bits. Each frame is then encrypted by xoring the first output bits of the keystream generator. The generator is initialized with a secret key, which remains the same during the whole session, and an IV which is modified for each new frame.

**Description of the running-key generator.** E0 is a combination generator composed of four LFSRs, combined by a Boolean function with a four-bit internal memory. This generator can be seen as a variant of the summation generator [Rue86].

This generator is used at two different levels: it is first applied during the initialization phase for generating a 128-bit initial state from the secret key and the IV. Then, the same mechanism is used to produce the keystream from the initial state.

The four LFSRs are binary LFSRs of respective lengths $L_1 = 25$, $L_2 = 31$, $L_3 = 33$, $L_4 = 39$ (i.e., a total of 128 bits) with feedback polynomials

$$\begin{aligned} P_1(x) &= x^{25} + x^{20} + x^{12} + x^8 + 1 \\ P_2(x) &= x^{31} + x^{24} + x^{16} + x^{12} + 1 \\ P_3(x) &= x^{33} + x^{28} + x^{24} + x^4 + 1 \\ P_4(x) &= x^{39} + x^{36} + x^{28} + x^4 + 1 \ . \end{aligned}$$

Let $x_t^i$ denote the output at time $t$ of the $i$-th LFSR. Then, the 3-bit integer (between 0 and 4) corresponding to the sum of the outputs of the four LFSRs is computed:

$$y_t = x_t^1 + x_t^2 + x_t^3 + x_t^4 = 4y_t^2 + 2y_t^1 + y_t^0$$

where $y_t^2, y_t^1, y_t^0$ are binary values. The generator also includes some internal memory composed of two 2-bit words, denoted by $c_t$ and $c_{t-1}$ at time $t$. If $2c_t^1 + c_t^0 = c_t$ denotes the binary decomposition of $c_t$, then this 2-bit word is updated by

$$\begin{aligned} c_{t+1}^1 &= z_{t+1}^1 + c_t^1 + c_{t-1}^0 \bmod 2 \\ c_{t+1}^0 &= z_{t+1}^0 + c_t^0 + c_{t-1}^1 + c_{t-1}^0 \bmod 2 \end{aligned}$$

where $z_t^1, z_t^0$ is the binary decomposition of the integer

$$z_t = \left\lfloor \frac{y_{t-1} + c_{t-1}}{2} \right\rfloor \ .$$

This combination generator with memory is directly used for producing the keystream: the keystream at time $t$ is equal to

$$s_t = y_t^0 + c_t^0 \bmod 2 \ .$$

The generator is initialized by the means of an additional level of the previously described system. The four LFSRs are first initialized with the 128-bit value

$$G_1(K_c) \text{ XOR } G_2(IV)$$

where $G_1$ and $G_2$ are two affine transformations with a 128-bit output. The two memory words are set to zero. Then, the generator is clocked 200 times, and another affine transformation $G_3$ is applied to the last 128 bits produced by the generator. The result of this operation then is then used as an initial state for the second-level generator, i.e., for the generator which outputs the keystream. The internal memory of this second-level generator is given by the memory of the first-level generator after the first 200 clocks.

Figure 3.12: E0 keystream generator.

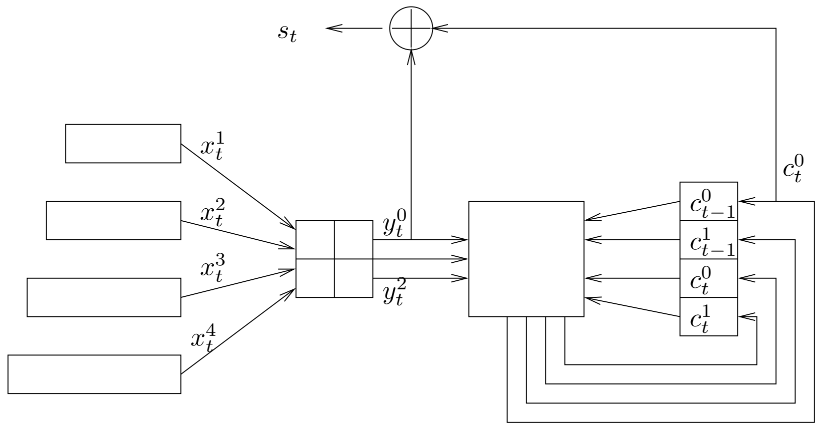
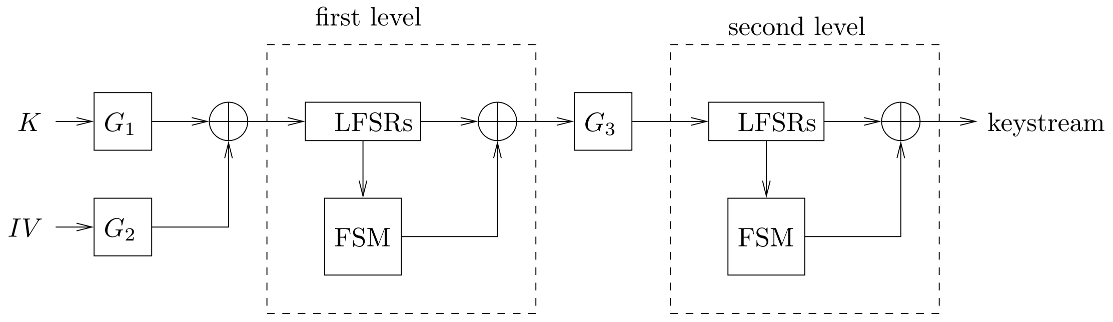


Figure 3.13: Initialization of E0 keystream generator.



**Attacks on E0.** The combination used in E0 is vulnerable to several attacks, including a linear attack [GBM02], some algebraic and fast algebraic attacks [Arm02, Cou03, HR04], and some fast correlation attacks [LV04b]. But all these attacks require the knowledge of a huge number of consecutive keystream bits generated from the same internal state, which is not the case in the Bluetooth protocol since the generator is resynchronized after 2745 bits.

Nevertheless, some sophisticated correlation attacks due to Yi Lu, Willi Meier and Serge Vaudenay [LV04a, LMV05] take the resynchronization process into account. The most efficient attack recovers the secret key from the knowledge of the first 24 bits of $2^{23.8}$ keystream frames. Its time complexity corresponds to $2^{38}$ operations. A better trade-off between the on-line computation and the precomputation has been obtained in [ZXF13]. All these results imply that the security level of the E0 stream cipher is very limited.

## 3.5 Correlation attacks on LFSR-based generators

The *correlation attack* was originally proposed by Siegenthaler in 1985 [Sie85] against the combination generator composed of $n$ LFSRs of lengths $L_1, \ldots, L_n$. The correlation attack is a divide-and-conquer technique: it aims at recovering the initial state of each constituent

LFSRs separately from the knowledge of some keystream bits (in a known plaintext attack). This attack requires $\sum_{i=1}^{n}\left(2^{L_i}-1\right)$ trials only, instead of the $\prod_{i=1}^{n}\left(2^{L_i}-1\right)$ tests required by an exhaustive search. A similar ciphertext only attack can also be mounted when there exists redundancy in the plaintext, as mentioned in [Sie85].

### 3.5.1 General principle

More generally, the correlation attack applies to any keystream generator as soon as the keystream is correlated to the output sequence $\boldsymbol{\sigma}$ of a "reduced generator" whose initial state depends on some key bits only. These key bits can be determined by recovering the initialization of $\boldsymbol{\sigma}$ as follows: an exhaustive search for the initialization of $\boldsymbol{\sigma}$ is performed, and the correct one is detected by computing the correlation between the corresponding sequence $\boldsymbol{\sigma}$ and the keystream. The following description concentrates on binary sequences.

More precisely, we assume as on Figure 3.14 that the internal state of the generator at time $t$ can be divided into two parts, $x_t$ and $y_t$ of respective sizes $\ell$ and $(n-\ell)$, which are updated independently by two functions $\Phi_0$ and $\Phi_1$. Suppose wlog. that the attacker aims at recovering the first part of the initial state, $x_0$. The input vector of the filtering function can be decomposed into two parts, $x$ and $y$ according to the previous decomposition. Then, the attack can be mounted if there exists a function $g$ depending on $\ell$ variables (i.e., depending on $x$ only) whose output coincides with the output of $f$ for more than half of the inputs. In other words, if there exists an $\ell$-variable function $g$ such that

$$p_g = \mathrm{Pr}_{X,Y}[f(X,Y)=g(X)] > \frac{1}{2}\ .$$

The existence of such a function $g$ and its optimal choice is discussed in Section 3.5.5.

If $p_g > 1/2$, then the sequence $\boldsymbol{\sigma}(x_0)$ produced by the reduced generator with initial state $x_0$ and filtering function $g$ is correlated to the keystream $\mathbf{s}$. Indeed, for all $t \geq 0$,

$$\Pr[s_t = \sigma_t] = p_g > \frac{1}{2}\ .$$

### 3.5.2 Correlation attacks as a decoding problem

It has been observed by Meier and Staffelbach [MS89] that the previously described situation corresponds to a classical problem in the context of error-correction. Indeed, if there exists a correlation between the keystream $\mathbf{s}$ and the output $\boldsymbol{\sigma}$ of the reduced generator, then the keystream subsequence $(s_t)_{t<N}$ can be seen as the result of the transmission of $(\sigma_t)_{t<N}$ through the binary symmetric channel with error probability $p = \Pr[s_t \neq \sigma_t] = 1 - p_g < 1/2$ (see Fig. 3.15). Moreover, if the transition function $\Phi_0$ of the reduced generator is linear, then all bits of $\boldsymbol{\sigma}$ depend linearly on $x_0$. Therefore, $(\sigma_t)_{t<N}$ is a codeword of a linear code $\mathcal{C}$ of length $N$ and dimension $\ell$ defined by $\Phi_0$. Thus, recovering the initial state $x_0$ consists in decoding the running-key subsequence relatively to this linear code.

From Shannon's theorem, we know that $(\sigma_t)_{t<N}$ can only be decoded without errors if the transmission rate of the code exceeds the capacity of the channel. The involved channel is the binary symmetric channel with cross-over probability $p = (1-p_g)$. The capacity of the channel is defined by

$$C(p) = 1 + p\log_2 p + (1-p)\log_2(1-p)\ .$$

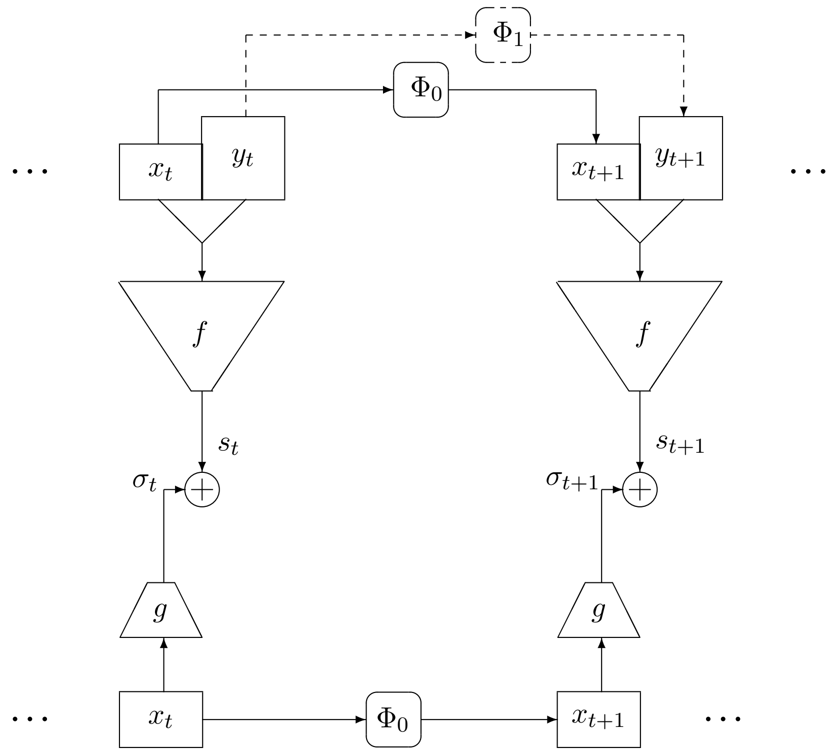


Figure 3.14: Model for the correlation attack.

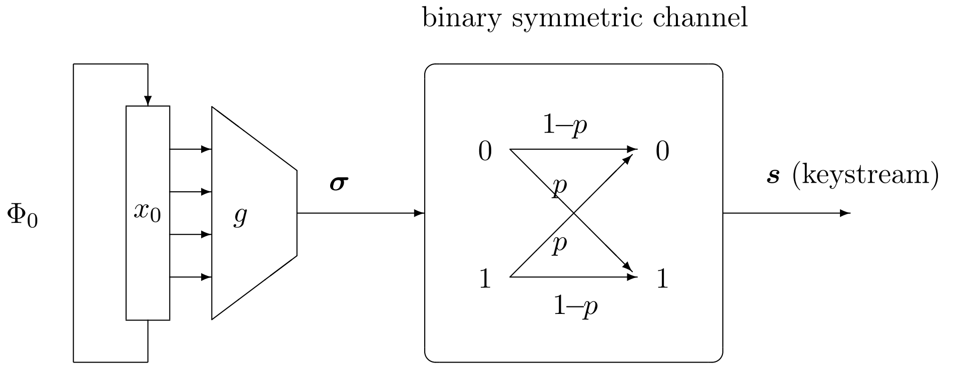


Figure 3.15: Correlation attacks on LFSR-based stream ciphers seen as a decoding problem.

In most situations, $p_g$ is close to 1/2, i.e., $p_g = 1/2(1+\varepsilon)$ with $\varepsilon \ll 1$. Then, we get the following approximation

$$\begin{aligned} C\left(\frac{1}{2}(1-\varepsilon)\right) &= 1 - \frac{1}{2\ln 2}\left[(1-\varepsilon)\ln\left(\frac{1-\varepsilon}{2}\right) + (1+\varepsilon)\ln\left(\frac{1+\varepsilon}{2}\right)\right] \\ &\simeq 1 - \frac{1}{2\ln 2}\left[(1-\varepsilon)(-\varepsilon) + (1+\varepsilon)\varepsilon - 2\ln(2)\right] \\ &= 1 + \frac{\varepsilon^2}{2\ln 2} - 1 = \frac{\varepsilon^2}{\ln 2} \,. \end{aligned}$$

The transmission rate of the code is equal to $\ell/N$ where $\ell$ is the size of the targeted part of the initial state $x_0$ and $N$ is the number of known keystream bits. Then, we deduce from Shannon's theorem that the attack requires the knowledge of

$$N \geq \frac{\ell \ln 2}{\varepsilon^2} \text{ keystream bits.} \tag{3.1}$$

**Generator matrix for an LFSR-code.** In the particular case where the internal state $x_t$ of the reduced generator is updated by an LFSR of length $\ell$, the generator matrix of the underlying code $\mathcal{C}$ can be easily computed from the characteristic polynomial of the LFSR.

**Proposition 3.8.** *Let $(\sigma_t)_{t\geq 0}$ be a sequence produced by an LFSR of length $\ell$ with characteristic polynomial $P^\star$. Then, for any $N \geq \ell$, the $\ell \times N$ matrix $G$ such that*

$$(\sigma_0, \dots, \sigma_{N-1}) = (\sigma_0, \dots, \sigma_{\ell-1})G$$

*is the matrix whose $t$-th column, $0 \leq t < N$ corresponds to the coefficients of the polynomial $X^t \bmod P^\star(X)$.*

*Proof.* Let $x_0$ denote the initial state of the reduced generator producing $(\sigma_t)_{t\geq 0}$. From Proposition 3.4, we know that, for any $t \geq 0$,

$$\sigma_t = \mathsf{Tr}(\alpha^t x_0) = \mathsf{Tr}\left(\alpha^t \left(\sum_{i=0}^{\ell-1} \sigma_i \beta_i\right)\right) .$$

where $\alpha$ is a root of $P^\star$ and $\{\beta_0, \dots, \beta_{\ell-1}\}$ is the dual basis of $\{1, \alpha, \dots, \alpha^{\ell-1}\}$. Let $X^t \bmod P^\star(X) = \sum_{j=0}^{\ell-1} d_j X^j$. Then, by definition of $\alpha$, we have that $\alpha^t = \sum_{j=0}^{\ell-1} d_j \alpha^j$. It follows that

$$\begin{aligned}
\sigma_t &= \mathsf{Tr}\left(\left(\sum_{j=0}^{\ell-1} d_j \alpha^j\right)\left(\sum_{i=0}^{\ell-1} \sigma_i \beta_i\right)\right) \\
&= \sum_{j=0}^{\ell-1}\sum_{i=0}^{\ell-1} d_j \sigma_i \mathsf{Tr}(\alpha^j \beta_i) \\
&= \sum_{i=0}^{\ell-1} d_i \sigma_i
\end{aligned}$$

by definition of the dual basis. This equivalently means that the $t$-th column of $G$ corresponds to the vector $(d_0, \dots, d_{\ell-1})$. ◇

### 3.5.3 Maximum-likelihood decoding for correlation attacks

The original correlation attack presented by Siegenthaler [Sie85] recovers the initial state $x_0$ by performing a statistical test on the correlation between the observed keystream $\mathbf{s}$ and the output of the reduced generator $\boldsymbol{\sigma}(x_0)$ for all possible values of $x_0$. This exactly corresponds to a maximum-likelihood decoding procedure. Here, we describe the attack in terms of statistical test as in the original paper.

**Proposition 3.9.** *Let $(s_t)_{t\geq 0}$ and $(\sigma)_{t\geq 0}$ be two sequences such that $\Pr[s_t \neq \sigma_t] = p$ for all $t \geq 0$. Then, the correlation between these two sequences computed over $N$ bits,*

$$C = \frac{1}{N}\sum_{t=0}^{N-1}(-1)^{s_t+\sigma_t}$$

*is a random variable which follows a Gaussian distribution with mean $(1-2p)$ and variance $4p(1-p)/N$.*

*Proof.* We write

$$C = \frac{1}{N}\sum_{t=0}^{N-1}(1-2(s_t \oplus \sigma_t)) = 1 - \frac{2}{N}\sum_{t=0}^{N-1}(s_t \oplus \sigma_t)\ .$$

The $N$ variables $(s_t \oplus \sigma_t)_{0\leq t<N}$ are independent and follow a Bernoulli distribution with mean $p$ and variance $p(1-p)$. Then, from the central limit theorem, we get that, for large $N$, the distribution of

$$\frac{1}{N}\sum_{t=0}^{N-1}(s_t \oplus \sigma_t)$$

is close to the normal distribution with mean $p$ and variance $p(1-p)/N$. Thus, $C$ follows the normal distribution with mean $(1-2p)$ and variance $4p(1-p)/N$. ◇

In particular, the previous proposition implies that, if the two sequences are uncorrelated, i.e., if $p = 1/2$, then $C$ follows a normal distribution with mean 0 and variance $1/N$.

If the two sequences $(\sigma_t)_{t\geq 0}$ and $(s_t)_{t\geq 0}$ are correlated, then all possible values for the initial state $x_0$ of $\boldsymbol{\sigma}$ are examined. The correct value for $x_0$ can be detected by a classical hypothesis testing (see Section 2.3.3 in Chapter 2). The corresponding algorithm is described in Algorithm 8. The value of the correlation is compared to some threshold $T$, whose value is

**Algorithm 8** Original correlation attack.

```
**Input.** $s_0s_1\dots s_{N-1}$, $N$ keystream bits and $p = \Pr[s_t \neq \sigma_t] < 1/2$.
**Output.** $\sigma_0\dots\sigma_{\ell-1}$, the initial state of $\boldsymbol{\sigma}$.
Compute the threshold $T$ with (3.2)
for all $\sigma_0,\dots,\sigma_{\ell-1}$ do
  Generate the first $N$ bits of the sequence $\boldsymbol{\sigma}$.
  Compute the correlation between $s_0s_1\dots s_{N-1}$ and $\sigma_0\sigma_1\dots\sigma_{N-1}$:
      $C \leftarrow \frac{1}{N}\sum_{t=0}^{N-1}(-1)^{s_t+u_t \bmod 2}$
  if $C > T$ then
    return $\sigma_0\dots\sigma_{\ell-1}$
  end if
end for
```

chosen as follows in order to minimize the error probability. If the initial state of $\boldsymbol{\sigma}$ is correct

(Hypothesis $\mathcal{H}_1$), then $C$ follows a normal distribution with mean $(1-2p)$ and variance $4p(1-p)/N$. This means that

$$\Pr[C=x] = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-(1-2p))^2}{2\sigma^2}\right)$$

with $\sigma^2 = 4p(1-p)/N$. Otherwise (Hypothesis $\mathcal{H}_0$), we have

$$\Pr[C=x] = \sqrt{\frac{N}{2\pi}} \exp\left(-\frac{Nx^2}{2}\right) .$$

For instance, these two distributions for $N=50$ and $p=1/4$ are plot on Figure 3.16.

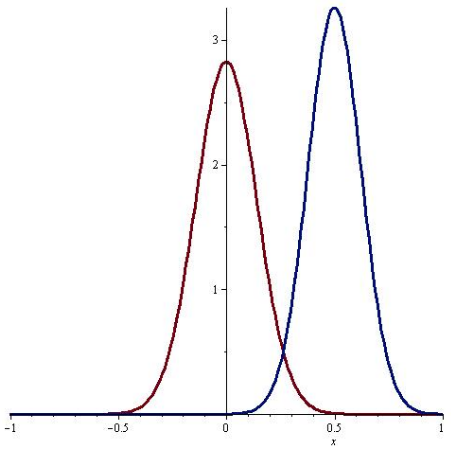


Figure 3.16: Distributions of the correlation for $N=50$ and $p=1/4$. The red curve corresponds to Hypothesis $\mathcal{H}_0$ (no correlation), while the blue curve corresponds to $\mathcal{H}_1$ (correct initial state).

Then, the false-alarm probability is

$$\begin{aligned} P_f &= \sqrt{\frac{N}{2\pi}} \int_T^{+\infty} \exp\left(-\frac{Nx^2}{2}\right) dx = \frac{1}{\sqrt{2\pi}} \int_{T\sqrt{N}}^{+\infty} \exp\left(-\frac{y^2}{2}\right) dy \\ &= 1 - \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{T\sqrt{N}} \exp\left(-\frac{y^2}{2}\right) dy = 1 - \Phi(T\sqrt{N}) \end{aligned}$$

where $\Phi(x)$ is the cumulative distribution function of the standard normal distribution, i.e.,

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp\left(-\frac{y^2}{2}\right) dy .$$

It is worth noticing that $\Phi$ can also be expressed by the means of the Gauss error function

$$\Phi(x) = \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right] .$$

Similarly, the non-detection probability is given by

$$\begin{aligned} P_n &= \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{T} \exp\left(-\frac{(x-(1-2p))^2}{2\sigma^2}\right) dx \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\frac{T-(1-2p)}{\sigma}} \exp\left(-\frac{y^2}{2}\right) dy = \Phi\left(\frac{T-(1-2p)}{\sigma}\right) . \end{aligned}$$

We deduce that, for obtaining a given value of $P_n$, we need to choose the threshold such that

$$\frac{T-(1-2p)}{\sigma} = \Phi^{-1}(P_n)$$

or equivalently

$$T = (1-2p) + 2\Phi^{-1}(P_n)\sqrt{\frac{p(1-p)}{N}} . \tag{3.2}$$

Moreover, we have

$$T\sqrt{N} = \Phi^{-1}(1-P_f)$$

implying that

$$\sqrt{N} = \frac{\Phi^{-1}(1-P_f) - 2\Phi^{-1}(P_n)\sqrt{p(1-p)}}{1-2p} .$$

Typical suitable values for $P_f$ and $P_n$ are $P_n = 1.3 \times 10^{-3}$, i.e., $\Phi^{-1}(P_n) = -3$ and $P_f = 2^{-\ell}$, i.e., $\Phi^{-1}(1-P_f) \simeq \sqrt{\ell}$. With these values, we get that

$$N = \mathcal{O}\left(\frac{\ell}{\varepsilon^2}\right) \text{ where } p = \frac{1}{2}(1-\varepsilon) .$$

It is worth noticing that we recover the data complexity deduced from Shannon's theorem and the value of the capacity of the binary symmetric channel. The time complexity of Algorithm 8 is then

$$\mathsf{Time} = 2^\ell N = \mathcal{O}\left(\frac{\ell 2^\ell}{\varepsilon^2}\right)$$

**Maximum-Likelihood decoding with an FFT.** The time complexity of the previous algorithm is prohibitive in most situations, except for very small values of $\ell$. However, it can be significantly reduced when the transition function $\Phi_0$ is linear (i.e., if the underlying code $\mathcal{C}$ is a linear code). Indeed, maximum-likelihood decoding consists in computing the distance between the $N$-bit received word (i.e., the keystream in our case) and the $2^\ell$ codewords in $\mathcal{C}$. When $\mathcal{C}$ is a linear code, this computation boils down to evaluating the Fourier transform of the ternary function $F$ from $\mathbb{F}_2^\ell$ into $\{-1,0,1\}$ defined by

$$\begin{cases} F(g_t) = (-1)^{s_t} & \text{for all } 0 \le t < N \\ F(x) = 0 & \text{for all } x \notin \{g_t,\ 0 \le t < N\} \end{cases}$$

where $g_t$ is the $t$-th column of the generator matrix of $\mathcal{C}$. Indeed, the correlation between the keystream and the sequence $\boldsymbol{\sigma}$ produced from a given initial state $x_0$ corresponds to the

Fourier transform of $F$ at point $x_0$:

$$\begin{aligned} C(\mathbf{s}, \boldsymbol{\sigma}(x_0)) &= \frac{1}{N}\sum_{t=0}^{N-1}(-1)^{s_t+\sigma_t} \\ &= \frac{1}{N}\sum_{t=0}^{N-1}(-1)^{s_t+x_0\cdot g_t} \\ &= \sum_{x\in\mathbb{F}_2^\ell} F(x)(-1)^{x_0\cdot x} = \widehat{F}(x_0)\ . \end{aligned}$$

Therefore, when the code length $N$ (i.e., the data complexity of the attack) is large, the time complexity can be reduced to $2^\ell \ell$ with a Fast Fourier Transform algorithm [CJM02, Lu06]. However, ML-decoding remains usually infeasible when the size of targeted part of the initial state, $x_0$, is not very small, typically when $\ell \geq 80$. In this situation, the attack can only be mounted by using some decoding algorithms faster than ML-decoding. The counterpart is of course that such algorithms require a longer keystream segment than the optimal value corresponding to Shannon's theorem (see Equation (3.1)). These variants of the original attack are usually named *fast correlation attacks*. They apply when the transition function of the reduced generator is linear.

### 3.5.4 Iterative decoding with based on low-density parity-checks

The idea of using an iterative decoding procedure based on low-density parity check (LDPC) equations comes back to Meier and Staffelbach [MS89], even if the algorithm they originally proposed was less efficient than the classical decoding algorithm for LDPC introduced by Gallager [Gal62]. The general principle of this decoding procedure consists in searching for a large number of parity-check relations for $\mathcal{C}$ having a low weight $w$. By the means of these relations, $\mathcal{C}$ can be seen as an LDPC code, for which there exist efficient decoding algorithms.

**Low-weight parity-check equations** Low-weight parity-check equations, i.e., with a small number of terms, for the involved code $\mathcal{C}$ can be exploited in fast correlation, but also in many other cryptanalytic techniques as they usually provide an efficient distinguishing procedure. A parity-check equation for the code $\mathcal{C}$ corresponds to a set of columns in the generator matrix which sum to zero. Equivalently, it can be seen as a codeword of the dual code $\mathcal{C}^\perp$.

If the transition function of the reduced generator corresponds to an LFSR, we deduce from the structure of the LFSR code, that its parity-check equations have the following form

$$\sigma_t + \sigma_{t+\tau_1} + \ldots + \sigma_{t+\tau_{w-1}} = 0, \text{ for all } t \geq 0$$

and are satisfied for any sequence $\boldsymbol{\sigma}$ generated by the LFSR. We know from Proposition 3.8 that the $t$-th column of the generator matrix of the LFSR code corresponds to the coefficients of $X^t \bmod P^\star(X)$, where $P^\star$ is the characteristic polynomial of the LFSR. Therefore, there is a bijection between such a parity-check equations and the polynomials

$$1 + X^{\tau_1} + \ldots + X^{\tau_{w-1}}$$

which are multiples of the LFSR characteristic polynomial $P^\star$. The weight of such an equation is the number of its terms, or equivalently the Hamming weight of the corresponding word in the dual code.

**Degree of parity-check equations.** The value $\tau_{w-1}$ (i.e., the degree of the corresponding polynomial) is also called the degree of the parity-check equation. For estimating the complexity of the attack, we need to determine the number of parity-check equations with a given weight $w$ and with a degree less than some value $d$, when $d$ varies. Usually, we assume that, for a given $w$, when $d$ is large enough, the values taken by the polynomials of weight $w$ of the form $(1 + X^{\tau_1} + X^{\tau_2} + \ldots + X^{\tau_{w-1}}) \bmod P^\star(X)$ for all $0 < \tau_1 < \tau_2 < \ldots < \tau_{w-1} < d$ are uniformly distributed in the set of all polynomials modulo $P^\star$ Under this hypothesis, the number of parity-check equations of weight $w$ and of degree at most $d$ is roughly

$$m_w(d) = \frac{\binom{d}{w-1}}{2^{\deg P^\star}} \simeq \frac{d^{w-1}}{(w-1)!\, 2^{\deg P^\star}} \ . \tag{3.3}$$

However, the previous hypothesis does not hold in all situations. It would imply for instance that, for any $P^\star$, the number of codewords of weight exactly $w$ in the dual of the LFSR code $\mathcal{C}^\perp$ of length $N = 2^{\deg P^\star} - 1$ is always close to

$$\frac{N^{w-1}}{w!} \ . \tag{3.4}$$

But clearly, this number highly depends on the algebraic structure of the characteristic polynomial. When $P^\star$ is a primitive polynomial, the corresponding LFSR code of length $N = 2^{\deg P^\star} - 1$ is equivalent to the *simplex code* (i.e., to the shortened first-order Reed-Muller code) [MS77, Page 30]. Its dual is then equivalent to the Hamming code of length $N$. In this case it can be checked that Formula (3.4) provides a good approximation of the weight distribution [MS77, Page 129]. More generally, when $P^\star$ is a randomly chosen primitive polynomial, simulation results show that (3.3) is a good estimation of the number of parity-check relations of weight $w$ and degree at most $d$, when $d$ is not too small. It is worth noticing that this holds even if $P^\star$ itself has weight $w$. Similarly, the minimal degree for a polynomial of weight $w$ multiple of $P^\star$ is close to

$$(w-1)!^{\frac{1}{w-1}} 2^{\frac{\deg P^\star}{w-1}} \ .$$

The situation is much more complicated when $P^\star$ is the product of two primitive polynomials. Then, the previous estimation remains reasonable when the degrees of the polynomials involved in the product are coprime, but does not hold in general. For instance, it is possible to construct some examples where $P^\star$ is the product of two primitive polynomials having the same degree and such that $P^\star$ has no multiple of degree 3 (see e.g. [CTZ01]).

**Computing low-degree parity-check equations.** There are several algorithms of computing parity-check equations of weight $w$ associated to a polynomial $P$. The simplest one is described in Algorithm 9. More sophisticated ones are mentioned in [Jou09, Pages 384-386] The corresponding time and memory complexity are

$$\mathsf{Time} = \mathcal{O}(d^{w-v-1}) \text{ and } \mathsf{Memory} = \mathcal{O}(d^v) \ .$$

An interesting trade-off is then obtained for $v = \lfloor \frac{w-1}{2} \rfloor$, leading to

$$\mathsf{Time} = \mathcal{O}(d^{\lceil \frac{w-1}{2} \rceil}) \text{ and } \mathsf{Memory} = \mathcal{O}(d^{\lfloor \frac{w-1}{2} \rfloor}) \ .$$

When $w \geq 5$, an improved variant of this algorithm due to Chose, Joux and Mitton [CJM02] allows to decrease the memory requirement to $\mathsf{Memory} = \mathcal{O}(d^{\lfloor \frac{w}{4} \rfloor})$ with the same time complexity.

**Algorithm 9** Algorithm for computing multiples of $P$ of weight $w$ and degree at most $d$.

```
for each set $(i_1, \dots, i_v)$ of $v$ elements of $\{1, \cdots, d\}$ do
  $q(X) \leftarrow X^{i_1} + \dots + X^{i_v} \bmod P(X)$
  store $q$ in a table $T$ such that $T[a] = \{(i_1, \dots, i_v) : q(X) = a\}$.
end for
for each set $(j_1, \dots, j_{w-v-1})$ of $w - v - 1$ elements of $\{1, \cdots, d\}$ do
  $A \leftarrow 1 + X^{j_1} + \dots + X^{w-v-1} \bmod P(X)$
  for all $(i_1, \dots, i_v) \in T[A]$ do
    return $1 + X^{i_1} + \dots + X^{i_v} + X^{j_1} + \dots + X^{w-v-1}$
  end for
end for
```

**Iterative decoding of LDPC.** A detailed comparison between several iterative decoding algorithms can be found in [Lev04]. For instance, the slightly impaired but simple version of Gallager's algorithm proposed in [CT00] consists in iteratively updating the log-likelihood ratio at each bit position, i.e., the quantity

$$\log\left(\frac{\Pr[\sigma_t = 0]}{\Pr[\sigma_t = 1]}\right) .$$

It is described by Algorithm 10.

**Algorithm 10** Iterative decoding algorithm for fast correlation attacks.

```
Input: $s_0 s_1 \dots s_{N-1}$, $N$ keystream bits and $p = \Pr[s_t \neq \sigma_t] < 1/2$.
/* Initialization */
for all $t$ from 0 to $N-1$ do
  $L[t] \leftarrow \log(\frac{1-p}{p})$
end for
repeat
  for all $t$ from 0 to $N-1$ do
    $L'[t] \leftarrow (-1)^{s_t} L[t]$
    for all parity-check equations involving Position $t$, $\sigma_t = \sum_{j \in J} \sigma_j$ do
      $$L'[t] \leftarrow L'[t] + (-1)^{\sum_{j \in J} s_j} \min_{j \in J}(L[j])$$
    end for
  end for
until convergence
for all $t$ from 0 to $N-1$ do
  if $L'[t] < 0$ then
    $\sigma_t \leftarrow 0$
  else
    $\sigma_t \leftarrow 1$
  end if
end for
return $(\sigma)_{t \geq 0}$
```

The complexity of this algorithm highly depends on the number $m_w$ of parity-check equations of weight $w$ required for convergence. The simulation results presented in [CT00] show that the minimal number of equations can be estimated by

$$m_w = \frac{2 \ln 2}{\varepsilon^{2w-4}} ,$$

where $\varepsilon = 1 - 2p$ and $p$ is the error-probability. By combining this result with the expected number of parity-check equations of weight $w$ and degree at most $N$ given by (3.3), we deduce that the required data complexity is

$$N = \left(\frac{1}{\varepsilon}\right)^{\frac{2(w-2)}{w-1}} 2^{\frac{\ell}{w-1}}$$

and the corresponding time complexity

$$\mathsf{Time} = \left(\frac{1}{\varepsilon}\right)^{\frac{2w(w-2)}{w-1}} 2^{\frac{\ell}{w-1}} .$$

### 3.5.5 Existence of an approximation with fewer variables

As explained in Section 3.5.1, a (fast) correlation attack can be mounted only when the Boolean function $f$ of $n$ variables involved in the keystream generator can be approximated by a function $g$ depending on fewer variables. Typically, in a combination generator composed of $n$ LFSRs, if the combining function $f$ can be approximated by a function $g$ of $\ell < n$ variables, then the attack involves the initial states of only $\ell$ out of the $n$ constituent LFSRs.

**Correlation-immunity order.** A natural counter-measure to avoid correlation attacks consists then in using as building-blocks a *correlation-immune function* in the sense of the following definition.

**Definition 3.10** (Correlation-immunity [Sie84])**.** *A Boolean function $f$ is* $t$-th order correlation-immune *if the probability distribution of its output is unaltered when any $t$ input variables are fixed. Balanced $t$-th order correlation-immune functions are called* $t$-resilient*.*

Note that a $t$-th order correlation-immune function is $k$-th order correlation-immune for any $k \leq t$. The correlation-immunity order of a function $f$ then refers to the highest integer $t$ such that $f$ is $t$-th order correlation-immune.

In a combination generator, the correlation-immunity order $t$ of the combining function determines the minimal number of LFSRs which must be considered together in a correlation attack. Indeed, the keystream produced by the combination generator is then independent of any set of $t$ constituent LFSRs. The smallest number of LFSRs involved in a correlation attack is therefore $t + 1$. But the correlation-immunity order of a Boolean function cannot be chosen as high as we wish: it is limited by the algebraic degree of the function as shown in the next proposition.

**Proposition 3.11.** *[Sie84] Let $f$ be a Boolean function of $n$ variables. Then its correlation-immunity order $t$ satisfies*

$$t + \deg f \leq n .$$

*Moreover, if $f$ is balanced and $t < n - 1$, then*

$$t + \deg f \leq n - 1 .$$

*Proof.* Let $u \in \mathbb{F}_2^n$ such that $wt(u) \geq n-t$. We compute the coefficients of the ANF of $f$ with Theorem 1.3. For $L = \{1,\dots,n\} \setminus \mathsf{Supp}(u)$, we have

$$a_u = \sum_{x_i=0, i\in L} f(x_1,\dots,x_n) \bmod 2 = 2^{-(n-wt(u))} wt(f) \bmod 2$$

where the last equality comes from the fact that the probability distribution of the output of $f$ is unchanger when the $(n - wt(u)) \leq t$ variables defined by $L$ are set to 0. If $f$ is balanced, i.e., $wt(f) = 2^{n-1}$, we get that, for all $u$ with $wt(u) \geq n-t$,

$$a_u = 2^{wt(u)-1} \bmod 2 = 0$$

since $wt(u) - 1 \geq n-t-1 > 0$. In other words, all coefficients of degree greater than or equal to $(n-t)$ in the ANF of $f$ are equal to zero, which means than $\deg f < n-t$.

If $f$ is not balanced, we select some word $u^\star$ of weight $(n-t)$. Then,

$$a_{u^\star} = 2^{-t} wt(f) \bmod 2$$

implying that $wt(f) = 2^t a_{u^\star} + \Lambda 2^{t+1}$ for some integer $\Lambda$. Now, for any $u$ of weight $(n-t+w)$ with $w \geq 1$, we get

$$a_u = 2^{-t+w} wt(f) = 2^w a_{u^\star} + \Lambda 2^{w+1} = 0 \bmod 2 \ .$$

This means that all coefficients in the ANF of degree greater than or equal to $(n-t+1)$ vanish, i.e. $\deg f \leq n-t$. ◇

**Approximation of a function by a function of $(t+1)$ variables.** Since the combining function in a combination generator is balanced and nonlinear, its correlation-immunity order is at most $(n-3)$. It follows that we can always apply a correlation attack by considering at most $\ell = (n-2)$ LFSRs together. We now show how to determine the best approximation of $f$ by a Boolean function $g$ depending on a fixed subset of $\ell$ variables.

**Proposition 3.12.** *[Can02, Zha00] Let $f$ be a function of $n$ variables and let $L$ be a subset of $\{1,\dots,n\}$ of cardinality $\ell$, $L = \{i_1,\dots,i_\ell\}$. The highest possible value over all $\ell$-variable functions $g$ of*

$$p_g = \Pr_X[f(X_1,\dots,X_n) = g(X_{i_1},\dots,X_{i_\ell})]$$

*is achieved if and only if*

$$\begin{cases} g(x) = 1 & \text{if } \Pr_Y[f(X,Y) = 1 | X = x] > \frac{1}{2} \\ g(x) = 0 & \text{if } \Pr_Y[f(X,Y) = 1 | X = x] < \frac{1}{2} \end{cases}$$

*It follows that the maximum of $p_g$ is*

$$\max_g p_g = \frac{1}{2} + \frac{1}{2^\ell} \sum_{x \in \mathbf{F}_2^\ell} \left| \frac{1}{2} - \Pr_Y[f(X,Y) = 1 | X = x] \right| \ .$$

*Proof.* Let us decompose the $n$ input variables of $f$ as $(X,Y)$ where $X$ corresponds to the $\ell$ variables of index $i_1,\dots,i_\ell$, and $Y$ to the $(n-\ell)$ remaining variables. Let $p_L(x)$, $x \in \mathbb{F}_2^\ell$, denote the probability $p_L(x) = \Pr_Y[f(X,Y) = 1 | X = x]$. In other words, $p_L(x)$ is the probability that $f$ outputs 1 when the $\ell$ inputs in positions $L$ are fixed and equal to $x$. For any $\ell$-variable $g$ we have

$$\begin{aligned} p_g &= \Pr_{X,Y}[f(X,Y) = g(X)] \\ &= 2^{-\ell}\left(\sum_{x\in g^{-1}(0)} \Pr[f(X,Y)=0|X=x] + \sum_{x\in g^{-1}(1)} \Pr[f(X,Y)=1|X=x]\right) \\ &= 2^{-\ell}\sum_{x\in g^{-1}(0)}(1-p_L(x)) + 2^{-\ell}\sum_{x\in g^{-1}(1)} p_L(x) \\ &= 2^{-\ell}|g^{-1}(0)| - 2^{-\ell}\sum_{x\in\mathbb{F}_2^\ell}(-1)^{g(x)}p_L(x) \\ &= \frac{1}{2} + 2^{-\ell-1}\sum_{x\in\mathbb{F}_2^\ell}(-1)^{g(x)} - 2^{-\ell}\sum_{x\in\mathbb{F}_2^\ell}(-1)^{g(x)}p_L(x) \\ &= \frac{1}{2} + 2^{-\ell}\sum_{x\in\mathbb{F}_2^\ell}(-1)^{g(x)}\left(\frac{1}{2} - p_L(x)\right) . \end{aligned}$$

It follows that $p_g$ is maximal if and only if all terms in the sum are greater than or equal to zero, or equivalently

$$g(x) = \begin{cases} 0 & \text{if } p_L(x) < 1/2, \\ 1 & \text{if } p_L(x) > 1/2 . \end{cases} \tag{3.5}$$

Note that the value of $g(x)$ can be arbitrarily chosen when $p_L(x) = \frac{1}{2}$. The maximal value of $p_g$ directly follows. ◇

The previous proposition obviously implies that the maximal value of $p_g$ is $1/2$ if $\ell$ is less than or equal to the correlation-immunity order of $f$ since all $\Pr_Y[f(X,Y) = 1|X = x] = 1/2$ in this case. Another consequence is that, for $\ell = t + 1$ where $t$ is the correlation-immunity order of $f$, the maximal value of $p_g$ is achieved by an affine function.

**Theorem 3.13.** *[CT00] Let $f$ be a $t$-resilient function of $n$ variables and let $L$ be a subset of $\{1, \ldots, n\}$ of cardinality $(t + 1)$, $L = \{i_1, \ldots, i_{t+1}\}$. The highest possible value over all $(t + 1)$-variable functions $g$ of*

$$p_g = \Pr_X[f(X_1, \ldots, X_n) = g(X_{i_1}, \ldots, X_{i_{t+1}})]$$

*is achieved by the affine function*

$$g(x_{i_1}, \ldots, x_{i_{t+1}}) = \sum_{i\in L} x_i + \varepsilon$$

*with $\varepsilon \in \mathbb{F}_2$.*

*Proof.* We here use the same notation as in the proof of the previous proposition. For any $j \in L$, $e_j$ denotes the $(t+1)$-bit vector whose all coordinates are zero except the $j$-th one. We first prove that, for any $x \in \mathbb{F}_2^{t+1}$ and any $j \in L$, $p_L(x) + p_L(x + e_j) = 1$. Indeed, we have

$$\begin{aligned} p_L(x) + p_L(x+e_j) &= \Pr[f(X,Y) = 1|X = x] + \Pr[f(X,Y) = 1|X = x + e_j] \\ &= 2\,(\Pr[f(X,Y) = 1|\forall i \in L, X_i = x_i]\Pr[X_j = x_j] + \\ &\quad \Pr[f(X,Y) = 1|\forall i \in L \setminus \{j\}, X_i = x_i, X_j \neq x_j]\Pr[X_j \neq x_j]) \\ &= 2\Pr[f(X,Y) = 1|\forall i \in L \setminus \{j\}, X_i = x_i] = 1 \end{aligned}$$

where the last equality comes from the fact that $f$ is $t$-resilient and that the set $L \setminus \{j\}$ has cardinality $t$. Let $g$ be a $(t+1)$-variable function such that $p_g$ is minimal. Since $p_L(x) + p_L(x+e_j) = 1$ for any $x \in \mathbb{F}_2^{t+1}$ and for any $j \in L$, Condition (3.5) implies that

$$g(x) + g(x+e_j) = 1$$

when $p_L(x) \neq \frac{1}{2}$. Moreover, we can assume that this relation is satisfied for any $x \in \mathbb{F}_2^{t+1}$, because the value of $g(x)$ can be arbitrarily chosen when $p_L(x) = \frac{1}{2}$. It follows that, for any $x \in \mathbb{F}_2^{t+1}$,

$$g(x) = g(0) + \sum_{i \in L} x_i \ .$$

This probability is then maximized when $(-1)^{g(0)}$ and $\left(\Pr_X[f(X) = \sum_{i \in L} X_i] - \frac{1}{2}\right)$ have the same sign. $\diamond$

This result is of great interest because the degree of the approximation $g$ affects the linear complexity of the reduced generator. Indeed, $\boldsymbol{\sigma}$ is produced by a smaller combining generator composed of $\ell = t+1$ (or more) LFSRs combined by $g$. Then, we know from Section 3.3.1 that the linear complexity of $\boldsymbol{\sigma}$ depends on the degree of $g$. The previous result then shows that, in a fast correlation attack involving the smallest number of LFSRs, i.e. $(t+1)$, the same combining function $g$ minimizes both the error probability $(1-p_g)$ and the linear complexity of $\boldsymbol{\sigma}$. In this context, since $g$ has degree 1, the minimal polynomial of $\sigma$ is the least common multiple of the minimal polynomials $P_i^\star$ of the considered LFSRs [Zie59]. In most practical situations, we have $P^\star = \prod_{j=1}^{t+1} P_{i_j}^\star$ since all the involved feedback polynomials are primitive. The $N$-bit keystream $(s_t)_{t<N}$ can then be seen as the result of the transmission through a noisy channel of a codeword of a linear code of dimension $\sum_{j=1}^{t+1} L_{i_j}$. The previously described decoding algorithm then applies and the analysis of its complexity tends to show that designing a combination generator which provides a reasonnable security is a very difficult (or even impossible) task.

**Correlation-immunity order and dual distance of a code.** The correlation-immunity order of a Boolean function $f$ can be characterized by a combinatorial property of the code formed by the support of $f$, i.e., by $f^{-1}(1)$. This combinatorial property corresponds to the notion of *orthogonal array* [Rao47].

**Proposition 3.14.** *[CCCS92] Let $f$ be a Boolean function of $n$ variables and let $\mathcal{C}_f$ be the code corresponding to its support, i.e., $\mathcal{C}_f = \{x \in \mathbb{F}_2^n : f(x) = 1\}$. Then, $f$ is $t$-th order correlation immune if and only if any set of $t$ columns in $\mathcal{C}_f$ contains the same number of occurences of every $t$-tuple.*

*Moreover, if $\mathcal{C}_f$ is a linear code, then $f$ is $t$-th order correlation immune if and only the minimum distance of $\mathcal{C}_f^\perp$ is strictly greater than $t$.*

*Proof.* By definition $\mathcal{C}_f$ has cardinality $wt(f)$. Let us consider any subset $T \subset \{1, \dots, n\}$ of size $t$. Then, $f$ is $t$-order correlation-immune if and only if, for any fixed value $a$ of the input variables at the positions defined by $T$, the set $\{f(x), x \in \mathbb{F}_2^n \text{ and } x_i = a_i \forall i \in T\}$ contains exactly $2^{-t}wt(f)$ ones. This equivalently means hat $\mathcal{C}_f$ contains exactly $2^{-t}wt(f)$ words which are equal to $a$ at the positions defined by $T$.

If $\mathcal{C}_f$ is a linear code and $G$ denotes a $k \times n$ generator matrix of $\mathcal{C}_f$, then the previous condition means that, for any $k \times t$ submatrix $G_T$ of $G$, the linear system $mG_T = a$ has $2^{k-\ell}$

solutions for any $a$. This is equivalent to the fact that any $k \times t$ submatrix $G_T$ of $G$ has rank $t$, i.e., any set of $t$ columns of $G$ are linearly independent. Now, we show that this condition is equivalent to the fact that the minimum distance of $\mathcal{C}_f^{\perp}$ is greater than $t$. Indeed $\mathcal{C}_f^{\perp}$ contains a codeword $x$ if and only if $Gx = 0$, i.e., the $wt(x)$ columns of $G$ corresponding to the support of $x$ sum to zero. In other words, the minimum distance of $\mathcal{C}_f^{\perp}$ is greater than or equal to $d^{\perp}$ if and only if any set of $w < d^{\perp}$ columns of $G$ are linearly independent. $\diamond$

The previous proposition shows that correlation-immunity order of a linear function is determined by the minimum distance of the dual of the code corresponding to its support. A very nice property is that this result holds even if the underlying code is not linear. In this situation, the *dual distance* can be defined as follows.

**Definition 3.15** (Dual distance of a code [Del73]). *Let $\mathcal{C}$ be a binary code of length $n$ and size $M$, and $(A_0, \dots, A_n)$ be its distance distribution, i.e.,*

$$A_i = \frac{1}{M} |\{(x, y) \in \mathcal{C} \times \mathcal{C}, d(x, y) = i\}| \ .$$

*Let $(A'_0, \dots, A'_n)$ be the vector obtained by applying the MacWilliams transform:*

$$\sum_{i=0}^{n} A'_i X^{n-i} Y^i = \frac{1}{M} \sum_{i=0}^{n} A_i (X + Y)^{n-i} (X - Y)^i \ .$$

*The* dual distance *of $\mathcal{C}$ is the smallest non-zero integer $i$ such that $A'_i \neq 0$.*

Obviously, when $\mathcal{C}$ is linear, its dual distance corresponds to the minimum distance of the dual code. Now, we will show that, even in the nonlinear case, the dual distance of $\mathcal{C}_f$ is related to the correlation-immunity of $f$. We will need the following lemma.

**Lemma 3.16.** *Let $\mathcal{C}$ be a binary code of length $n$. Then any set of $t$ columns in $\mathcal{C}$ contains the same number of occurences of every $t$-tuple if and only if for any $y \in \mathbb{F}_2^n$ with $1 \leq wt(y) \leq t$,*

$$\sum_{x \in \mathcal{C}} (-1)^{x \cdot y} = 0 \ .$$

*Proof.* Let us first consider some $T \subset \{1, \dots, n\}$ of size $t$ and some nonzero $y \in \mathbb{F}_2^n$ such that $\mathsf{supp}(y) \subset T$. For any $a \in \mathbb{F}_2^t$, we denote by $N_T(a)$ the number of codewords in $\mathcal{C}$ which are equal to $a$ at the positions defined by $T$. Then, we have

$$\sum_{x \in \mathcal{C}} (-1)^{x \cdot y} = \sum_{a \in \mathbb{F}_2^{\ell}} (-1)^{a \cdot y_T} N_T(a) \ . \tag{3.6}$$

where $y_T$ denotes the restriction of $y$ to the positions in $T$. Now, we suppose that, for any $T$ of size $t$, we have $N_T(a) = M 2^{-t}$ for all $a$. We consider any nonzero $y$ of weight at most $t$ and we choose $T$ of size $t$ such that $\mathsf{supp}(y) \subset T$. Then,

$$\sum_{x \in \mathcal{C}} (-1)^{x \cdot y} = \frac{M}{2^t} \left( \sum_{a \in \mathbb{F}_2^{\ell}} (-1)^{a \cdot y_T} \right) = 0 \ ,$$

since $y_T \neq 0$. Conversely, we consider any subset $T \subset \{1, \dots, n\}$ of size $t$. We deduce from (3.6) that, for any $\beta \in \mathbb{F}_2^t$

$$\begin{aligned}\sum_{y_T \in \mathbb{F}_2^t} (-1)^{\beta \cdot y_T} \left( \sum_{x \in \mathcal{C}} (-1)^{x \cdot (y_T, 0)} \right) &= \sum_{y_T \in \mathbb{F}_2^t} (-1)^{\beta \cdot y_T} \sum_{a \in \mathbb{F}_2^t} (-1)^{a \cdot y_T} N_T(a) \\ &= \sum_{a \in \mathbb{F}_2^t} N_T(a) \left( \sum_{y_T \in \mathbb{F}_2^t} (-1)^{(a+\beta) \cdot y_T} \right) = 2^t N_T(\beta) \ ,\end{aligned}$$

by using that $\sum_{y_T \in \mathbb{F}_2^t} (-1)^{(\alpha+\beta) \cdot y_T} = 0$ except for $\alpha + \beta = 0$. Then, if all $\left(\sum_{x \in \mathcal{C}} (-1)^{x \cdot y}\right)$ vanish when $wt(y) \leq t$ except for $y = 0$, we get that, for all $\beta$.

$$2^t N_T(\beta) = \sum_{x \in \mathcal{C}} (-1)^{x \cdot 0} = M \ .$$

Both properties are then equivalent. ◇

A direct corollary of the previous lemma is the following characterization of correlation-immune functions in terms of Walsh transform [XM88].

**Corollary 3.17.** *Let $f$ be a Boolean function of $n$ variables. Then, $f$ is $t$-th order correlation-immune if and only if for any $y$ such that $1 \leq wt(y) \leq t$,*

$$\sum_{x \in \mathbb{F}_2^n} (-1)^{f(x) + x \cdot y} = 0 \ .$$

*Proof.* This is derived from the previous lemma by using that for any $y \neq 0$

$$\sum_{x \in \mathbb{F}_2^n} (-1)^{f(x) + x \cdot y} = \sum_{x \in f^{-1}(0)} (-1)^{x \cdot y} - \sum_{x \in f^{-1}(1)} (-1)^{x \cdot y} = -2 \sum_{x \in f^{-1}(1)} (-1)^{x \cdot y} \ .$$

◇

Now, we can prove that the correlation-immunity order of a Boolean function is related to the dual distance of $\mathcal{C}_f = f^{-1}(1)$.

**Theorem 3.18** (Correlation-immunity order and dual distance [BGS94, SM95])**.** *Let $f$ be a Boolean function of $n$ variables and let $\mathcal{C}_f$ be the code corresponding to its support, i.e., $\mathcal{C}_f = \{x \in \mathbb{F}_2^n : f(x) = 1\}$. Then, $f$ is $t$-th order correlation immune if and only if the dual distance of $\mathcal{C}_f$ is strictly greater than $t$.*

*Proof.* By combining Proposition 3.14 and Lemma 3.16, we see that we need to show that $\mathcal{C}_f$ has dual distance $d^\perp$ if and only if any nonzero $y$ of weight at most $(d^\perp - 1)$ satisfies

$$\sum_{x \in \mathcal{C}_f} (-1)^{x \cdot y} = 0 \ .$$

The MacWilliams identity can be written by means of Krawtchouk polynomials

$$A'_i = 2^{-k} \sum_{j=0}^{n} A_j P_i(j), \ \forall 0 \leq i \leq n \ ,$$

with

$$P_k(i) = \sum_{j=0}^{k} (-1)^j \binom{i}{j} \binom{n-i}{k-j} .$$

We first prove a well-known property of Krawtchouk polynomials: for any $x \in \mathbb{F}_2^n$ with $wt(x) = i$,

$$\sum_{y \in \mathbb{F}_2^n, wt(y)=j} (-1)^{x \cdot y} = P_j(i) .$$

Let $I$ denote the support of $x$. Since $x \cdot y$ corresponds to the size of the intersection between the supports of $x$ and $y$, we get

$$\sum_{y \in \mathbb{F}_2^n, wt(y)=j} (-1)^{x \cdot y} = \sum_{J \subset \{1,\ldots,n\}, |J|=j} (-1)^{|I \cap J|} = \sum_{\ell=0}^{j} (-1)^\ell N_I(\ell)$$

where

$$N_I(\ell) = |\{J \subset \{1, \ldots, n\} \text{ avec } |J| = j : |I \cap J| = \ell\}| = \binom{i}{\ell} \binom{n-i}{j-\ell} .$$

We deduce that

$$\sum_{y \in \mathbb{F}_2^n, wt(y)=j} (-1)^{x \cdot y} = \sum_{\ell=0}^{j} (-1)^\ell \binom{i}{\ell} \binom{n-i}{j-\ell} = P_j(i) .$$

Let us now consider some integer $w$, $1 \le w \le n$. Let us define the matrix $M_w$ of size $M \times \binom{n}{w}$ whose rows are indexed by the words of $\mathcal{C}_f$ and whose columns are indexed by the $n$-bit words of weight $w$ by

$$(M_w)_{x,y} = (-1)^{x \cdot y} .$$

Multiplying $M_w$ by its transpose, we get that the coefficient of index $(x, x')$ of $M_w(M_w)^T$ is

$$\left(M_w(M_w)^T\right)_{x,x'} = \sum_{y \in \mathbb{F}_2^n, wt(y)=w} (-1)^{(x+x') \cdot y} = P_w(d(x, x')) .$$

Then, we deduce

$$\mathbf{1} \left(M_w M_w^T\right) \mathbf{1}^T = (\mathbf{1} M_w) (\mathbf{1} M_w)^T = \sum_{y \in \mathbb{F}_2^n, wt(y)=w} \left( \sum_{x \in \mathcal{C}_f} (-1)^{x \cdot y} \right)^2 .$$

Therefore,

$$\begin{aligned} \sum_{y \in \mathbb{F}_2^n, wt(y)=w} \left( \sum_{x \in \mathcal{C}_f} (-1)^{x \cdot y} \right)^2 &= \mathbf{1} \left(M_w M_w^T\right) \mathbf{1}^T \\ &= \sum_{x,x' \in \mathcal{C}_f} P_w(d(x, x')) = M \sum_{i=0}^{n} A_i P_w(i) = M^2 A'_w . \end{aligned}$$

It follows that $A'_w = 0$ if and only if $\sum_{x \in \mathcal{C}_f} (-1)^{x \cdot y} = 0$ for all $y \in \mathbb{F}_2^n$ with $wt(y) = w$. ⋄

## Bibliography

[Arm02] Frederik Armknecht. A linearization attack on the Bluetooth keystream generator. `http://th.informatik.uni-mannheim.de/people/armknecht/E0.ps`, 2002.

[BB06] Elad Barkan and Eli Biham. Conditional estimators: An effective attack on A5/1. In *Selected Areas in Cryptography - SAC 2005*, volume 3897 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2006.

[BBK08] Elad Barkan, Eli Biham, and Nathan Keller. Instant ciphertext-only cryptanalysis of GSM encrypted communication. *Journal of Cryptology*, 21(3):392–429, 2008.

[BD00] Eli Biham and Orr Dunkelman. Cryptanalysis of the A5/1 GSM stream cipher. In *Progress in Cryptology - Indocrypt 2000*, volume 1977 of *Lecture Notes in Computer Science*, pages 43–51. Springer-Verlag, 2000.

[Ber68] Elwyn R. Berlekamp. *Algebraic Coding Theory.* McGraw-Hill, 1968.

[BGS94] Jürgen Bierbrauer, K. Gopalakrishnan, and Douglas R. Stinson. Bounds for resilient functions and orthogonal arrays. In *Advances in Cryptology - CRYPTO'94*, volume 839 of *Lecture Notes in Computer Science*, pages 247–256, 1994.

[BGW99] Marc Briceno, Ian Goldberg, and David Wagner. A pedagogical implementation of A5/1. `http://jya.com/a51-pi.htm`, 1999.

[Blu] Bluetooth specifications – version 1.0b. `http://www.bluetooth.com`.

[Bry86] Lennart Brynielsson. On the linear complexity of combined shift register sequences. In *Advances in Cryptology - EUROCRYPT'85*, volume 219 of *Lecture Notes in Computer Science*, pages 156–160. Springer, 1986.

[BSW00] Alex Biryukov, Adi Shamir, and David Wagner. Real time cryptanalysis of A5/1 on a PC. In *Fast Software Encryption – FSE 2000*, volume 1978 of *Lecture Notes in Computer Science*, pages 1–19. Springer-Verlag, 2000.

[Can02] Anne Canteaut. On the correlations between a combining function and functions of fewer variables. In *IEEE Information Theory Workshop - ITW 2002*, pages 78–81, Bangalore, Inde, October 2002. IEEE Press.

[CCCS92] Paul Camion, Claude Carlet, Pascale Charpin, and Nicolas Sendrier. On correlation-immune functions. In *Advances in Cryptology - CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 86–100. Springer-Verlag, 1992.

[CF01] Anne Canteaut and Eric Filiol. Ciphertext only reconstruction of stream ciphers based on combination generators. In *Fast Software Encryption - FSE 2000*, volume 1978 of *Lecture Notes in Computer Science*, pages 165–180. Springer-Verlag, 2001.

[CJM02] Philippe Chose, Antoine Joux, and Michel Mitton. Fast correlation attacks: an algorithmic point of view. In *Advances in Cryptology - EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 209–221. Springer-Verlag, 2002.

[CKM94] Don Coppersmith, Hugo Krawczyk, and Yishay Mansour. The shrinking generator. In *Advances in Cryptology - CRYPTO'93*, volume 773 of *Lecture Notes in Computer Science*. Springer-Verlag, 1994.

[Cou03] Nicolas Courtois. Fast algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 176–194. Springer-Verlag, 2003.

[CT00] Anne Canteaut and Mickaël Trabbia. Improved fast correlation attacks using parity-check equations of weight 4 and 5. In *Advances in Cryptology - EUROCRYPT'2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 573–588. Springer-Verlag, 2000.

[CTZ01] Pascale Charpin, Aimo Tietäväinen, and Victor Zinoviev. On binary cyclic codes with codewords of weight 3 and binary sequences with the trinomial property. *IEEE Transactions on Information Theory*, 47(1):421–425, 2001.

[Del73] P. Delsarte. Four fundamental parameters of a code and their combinatorial signifiance. *Information and Control*, 23(5):407–438, décembre 1973.

[Dor87] Jean-Louis Dornstetter. On the equivalence between berlekamp's and euclid's algorithms. *IEEE Transactions on Information Theory*, 33(3):428–431, 1987.

[EJ03] Patrick Ekdahl and Thomas Johansson. Another attack on A5/1. *IEEE Transactions on Information Theory*, 49(1):284–289, 2003.

[Gal62] Robert G. Gallager. Low-density parity-check codes. *IRE Transactions on Information Theory*, IT-8:21–28, 1962.

[GBM02] Jovan Dj. Golic, Vittorio Bagini, and Guglielmo Morgari. Linear cryptanalysis of Bluetooth stream cipher. In *Advances in Cryptology - EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 238–255. Springer-Verlag, 2002.

[Gef73] Philip R. Geffe. How to protect data with ciphers that are really hard to break. *Electronics*, pages 99–101, 1973.

[GN95] Rainer Göttfert and Harald Niederreiter. On the Minimal Polynomial of the Product of Linear Recurring Sequences. *Finite Fields and Applications*, 1(2):204–218, 1995.

[Gol82] Solomon W. Golomb. *Shift register sequences.* Aegean Park Press, 1982.

[Gün88] Christoph G. Günther. Alternating Step Generators Controlled by De Bruijn Sequences. In *Advances in Cryptology - EUROCRYPT'87*, volume 304 of *Lecture Notes in Computer Science*, pages 5–14. Springer, 1988.

[Hel11] Tor Helleseth. Maximal-length sequences. In *Encyclopedia of Cryptography and Security, 2nd Ed.*, pages 763–766. Springer, 2011.

[Her86] Tore Herlestam. On functions of linear shift register sequences. In *Advances in Cryptology - EUROCRYPT '85*, volume 219 of *Lecture Notes in Computer Science*, pages 119–129. Springer-Verlag, 1986.

[HR04] Philip Hawkes and Gregory G. Rose. Rewriting variables: the complexity of fast algebraic attacks on stream ciphers. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 390–406. Springer-Verlag, 2004.

[Jou09] Antoine Joux. *Algorithmic Cryptanalysis.* Chapman & Hall/CRC, 2009.

[Key76] Edwin L. Key. An analysis of the structure and complexity of nonlinear binary sequence generators. *IEEE Transactions on Information Theory*, 22:732–736, 1976.

[KPPM12] Maria Kalendri, Dionisios Pnevmatikatos, Ioannis Papaefstathiou, and Charalampos Manifavas. Breaking the GSM A5/1 Cryptography Algorithm with Rainbow Tables and High-End FPGAs. In *Field Programmable Logic and Applications - FPL 2012*, pages 747–753. IEEE, Aug 2012.

[Lev04] Sabine Leveiller. *Quelques algorithmes de cryptanalyse du registre filtré.* PhD thesis, Ecole Nationale Supérieure des Télécommunications, Paris, 2004.

[LMV05] Yi Lu, Willi Meier, and Serge Vaudenay. The conditional correlation attack: A practical attack on Bluetooth Encryption. In *Advances in Cryptology - CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 97–117. Springer-Verlag, 2005.

[Lu06] Yi Lu. *Applied stream ciphers in mobile communications.* PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 2006.

[LV04a] Yi Lu and Serge Vaudenay. Cryptanalysis of Bluetooth keystream generator two-level E0. In *Advances in Cryptology - ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 483–499. Springer-Verlag, 2004.

[LV04b] Yi Lu and Serge Vaudenay. Faster correlation attack on Bluetooth keystream generator E0. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 407–425. Springer-Verlag, 2004.

[Mas69] James L. Massey. Shift-register synthesis and BCH decoding. *IEEE Transactions on Information Theory*, 15:122–127, janvier 1969.

[Mas01] James L. Massey. The Ubiquity of Reed-Muller Codes. In *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes - AAECC-14*, volume 2227 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2001.

[MJB04] Alexander Maximov, Thomas Johansson, and Steve Babbage. An improved correlation attack on A5/1. In *Selected Areas in Cryptography - SAC 2004*, volume 3357 of *Lecture Notes in Computer Science*, pages 1–18. Springer-Verlag, 2004.

[MS77] F. Jessie MacWilliams and Neil J.A. Sloane. *The theory of error-correcting codes.* North-Holland, 1977.

[MS89] W. Meier and O. Staffelbach. Fast correlation attack on certain stream ciphers. *Journal of Cryptology*, pages 159–176, 1989.

[MS95] Willi Meier and Othmar Staffelbach. The self-shrinking generator. In *Advances in Cryptology - EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 205–214. Springer-Verlag, 1995.

[NP09] Karsten Nohl and Chris Paget. GSM: SRSLY? In *26th Chaos Communication Congress - 26C3*, 2009.

[PFS00] Slobodan Petrović and Amparo Fúster-Sabater. Cryptanalysis of the A5/2 algorithm. Technical Report 2000/052, Cryptology ePrint Archive, 2000. `http://eprint.iacr.org/`.

[Rao47] Calyampudi Radhakrishna Rao. Factorial experiments derivable from combinatorial arrangements of arrays. *J. Roy. Statist.*, 9:128–139, 1947.

[RC10] Sondre Rønjom and Carlos Cid. Nonlinear equivalence of stream ciphers. In *Fast Software Encryption – FSE 2010*, volume 6147 of *Lecture Notes in Computer Science*, pages 40–54. Springer, 2010.

[RS87] Rainer A. Rueppel and Othmar Staffelbach. Products of linear recurring sequences with maximum complexity. *IEEE Transactions on Information Theory*, 33(1):124–131, 1987.

[Rue86] Rainer A. Rueppel. *Analysis and Design of stream ciphers.* Springer-Verlag, 1986.

[Sie84] Thomas Siegenthaler. Correlation-immunity of nonlinear combining functions for cryptographic applications. *IEEE Transactions on Information Theory*, IT-30(5):776–780, 1984.

[Sie85] Thomas Siegenthaler. Decrypting a class of stream ciphers using ciphertext only. *IEEE Transactions on Information Theory*, C-34(1):81–84, 1985.

[SM95] Douglas R. Stinson and James L. Massey. An infinite class of counterexamples to a conjecture concerning nonlinear resilient functions. *Journal of Cryptology*, 8(3):167–173, 1995.

[XM88] Guozheng Xiao and James L. Massey. A spectral characterization of correlation-immune combining functions. *IEEE Transactions on Information Theory*, IT-34(3):569–571, 1988.

[Zha00] Muxiang Zhang. Maximum correlation analysis of nonlinear combining functions in stream ciphers. *Journal of Cryptology*, 13(3):301–313, 2000.

[Zie59] Neal Zierler. Linear recurring sequences. *J. Soc. Indus. Appl. Math.*, 7:31–48, 1959.

[ZXF13] Bin Zhang, Chao Xu, and Dengguo Feng. Real Time Cryptanalysis of Bluetooth Encryption with Condition Masking. In *Advances in Cryptology - CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 165–182. Springer, 2013.