SECRETS OF LINEAR FEEDBACK SHIFT REGISTERS

DAVID A. SINGER

ABSTRACT. A Linear Feedback Shift Register (LFSR) is a device that can generate a long seemingly random sequence of ones and zeroes, which is important in cryptography. We consider the sometimes unexpected periodic properties of LFSRs, how to understand them using linear algebra, and how to relate them to finite fields, another important topic in cryptography. Along the way, we resolve the puzzle of what it means for a polynomial to be primitive.

1. INTRODUCTION

Modern cryptography comes in two flavors: *Private key*, or classical, crypto uses complicated substitution and transposition techniques to obscure a message and relies on the receiver sharing a secret key with the sender. *Public key* cryptography, an essential tool in internet security, uses powerful mathematical ideas to allow secure communication between parties who do not have a shared key.

Courses in the mathematics of cryptography attract at least two different groups of students: mathematics majors, many with strong backgrounds in algebra or number theory, and engineering students, with strong backgrounds in computer science and computer engineering.

For public-key cryptography, an understanding of finite fields is essential, and mathematics majors are likely to have the necessary background in field theory and linear algebra.

Engineering students routinely learn about Linear Feedback Shift Registers (LFSR's), which are used in communications and in generating pseudo-random sequences; students may even know how to physically build them. LFSR's can also be used to create extremely efficient private-key cryptosysems, although in their straightforward implementation they are not cryptographically secure. The students are taught that so-called maximal-length LFSR's employ *primitive polynomials*, which can be found in look-up tables, but the students don't know how a primitive polynomial works, or what happens if they are using

Date: June 7, 2020.

one that is not primitive. The explanation for this requires an understanding of finite fields.

So it is useful for both math majors and CS majors to learn the theory of finite fields and then apply this to the theory of LFSR's. The connection between the two topics is rather subtle. In fact, the question of what periodicity properties a not-necessarily-maximal n-bit LFSR may have does not seem to be addressed in the literature. In the last part of this article we illustrate the solution in the case of $n = 6$, which is just large enought to make the result interesting but small enough to allow for a complete solution.

2. Linear Feedback Shift Registers

A Linear Feedback Shift Register (LFSR) is a device that can generate a long seemingly random sequence of ones and zeroes; it is used in computer simulations of random processes, error-correcting codes, and other engineering applications. The ease with which shift registers can produce such sequences make them an attractive topic in an introductory course in the mathematics of cryptography.

A first course in cryptography inevitably explores the notion of the *One-Time Pad*. This cryptosystem, introduced by G. S. Vernam in 1917, is a "perfectly secure cryptosystem", that is, the cipher text does not leak any information about the message ([1], P. 53, [5], p. 336). It relies on generating long random sequences of letters or numbers.

Suppose the message M consists of a sequence $m_1 m_2 \dots m_l$ of l letters taken from the usual 26-letter English alphabet. The venerable *Caesar cipher* works by shifting each letter of the alphabet some fixed amount; the Vernam cipher shifts the letters by a different amount at each position in the message. To encrypt M, we generate a sequence $k_1 k_2 \dots k_l$ of l random letters of the alphabet, each letter chosen randomly and independently with uniform probability $\frac{1}{26}$. The ciphertext is then the sequence $c_1 c_2 \dots c_l$ with c_i determined by adding m_i to k_i, where we treat the letters as the integers from 0 to 25 $\mod 26$.

Mod 26 arithmetic is somewhat inconvenient, and mod 2 arithmetic is more natural in digital computers. So we assume that a message has been encoded in some standard way as a string $m_1 m_2 \dots m_l$ of "bits", i.e., zeroes and ones. The key is then a random binary string $k_1 k_2 \dots k_l$ and we compute the ciphertext by $c_i = m_i \oplus k_i$, where the operation $\oplus$ is addition mod 2. To decrypt, we use the simple formula $m_i = c_i \oplus k_i$. To do this, of course, the recipient of the message must have the key string. Since this string is completely arbitrary, it is theoretically impossible to recover the message without the key, since every possible message of length l can be encrypted to any ciphertext of length l.

There are some major practical difficulties with this scheme. First, the recipient must have previously received the key, which is as large as the message! Second, the key must be chosen completely randomly. To overcome these difficulties in practice, cryptographers try to come up with a device or algorithm for generating a long seemingly random binary string of bits using only a small random string S (called the "seed"). Then the sender and receiver only need to agree on the seed, which they can exchange using public-key cryptography.

By definition, this long string is *not* random, since we generate it algorithmically, but perhaps it simulates a random string in the sense of being unpredictable for someone who does not possess the seed; such a sequence is called "pseudo-random." This sequence should have statistical properties that true random sequences have; e.g., 0 and 1 should appear with roughly the same frequency, likewise the four strings of length two should each appear with roughly the same frequency, and so on. This is necessary but by no means sufficient for a secure cryptosystem. We need a *semantically secure* algorithm, so an adversary can not recover partial information about a message in a reasonable amount of time. As you might guess, this last item is one of the most challenging problems of modern crytography.

One simple and elegant (but definitely *not* cryptographically secure) algorithm, or *machine*, for generating a pseudo-random string is the LFSR, shown in Figure 1. It consists of n cells, each capable of storing one bit, either a 0 or 1. The device is controlled by a clock; at each time step it transfers the content of each cell the next cell. The last cell outputs its bit to the stream. To get the new content of the leftmost cell, we feed back the mod 2 sum of the contents of certain specified cells. Mathematically, this is expressed using *connection coefficients* c_i, one for each cell, each of which is 0 or 1. When $c_i = 1$ the content of the ith cell is added in to the feedback. The machine is exactly determined by these coefficients, as in formula (1) below. We will always assume $c_n = 1$, since otherwise we would get the same output by deleting the last cell. For each choice of the connection coefficients we get a machine, and there are 2^{n-1} possibilities (assuming $c_n = 1$), so there are 2^{n-1} different LFSR's with n cells.

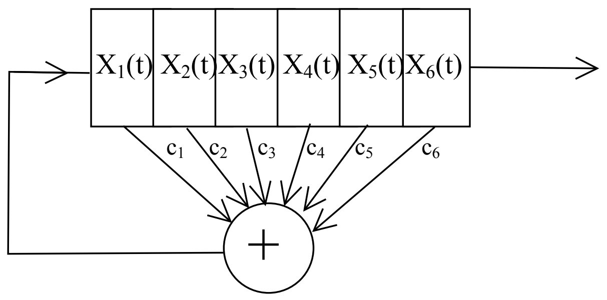

FIGURE 1. 6-cell LFSR

If $X_i(t)$ denotes the content of the ith cell at time step t, then the rules for the cells are

$$\begin{aligned} X_i(t+1) &= X_{i-1}(t) \qquad (2 \le i \le n) \\ X_1(t+1) &= c_1X_1(t) \oplus c_2X_2(t) \oplus \cdots \oplus c_nX_n(t) \end{aligned} \tag{2.1}$$

The state of the system at time t is given by the *column* vector

$$x(t) = \begin{pmatrix} X_1(t) \\ X_2(t) \\ \cdots \\ X_n(t) \end{pmatrix}.$$

The initial configuration is

$$x(0) = \begin{pmatrix} X_1(0) \\ X_2(0) \\ \cdots \\ X_n(0) \end{pmatrix} = \begin{pmatrix} s_1 \\ s_2 \\ \cdots \\ s_n \end{pmatrix} = s$$

where the initial state of the system is given by the seed s.

Since there are only 2^n possible states for the machine, it is obvious that whatever initial state is specified, the machine must eventually repeat. It is not quite as obvious that it must return to its initial state, and if we did not assume $c_n = 1$ this would not be true! For instance, the initial state $(0, 0, \ldots, 1)$ would drop into the zero state. We will soon see that if $c_n = 1$, the machine will always return to its initial state. If the initial state is the zero vector, then the machine will remain in that state forever, so we exclude that from consideration. Consequently, the maximum number of steps before the machine returns to its initial state is $2^n - 1$. Given a seed s, the *period* of s is the number of steps it takes to return to s; the period is the smallest positive r such that $x(r) = s = x(0)$. The period of the machine is the maximum period achieved for any seed. If the period of s is $2^n - 1$, then the machine must visit every non-zero state, and so the period for any seed must be $2^n - 1$; call such a machine a *maximal machine.* A fundamental fact in the theory of LFSR's is the fact that for every n a maximal machine exists. Since the goal is to achieve a long string from a small seed, this is the preferred result.

This is where the subject becomes mysterious. Of the 2^{n-1} possible machines, which choices correspond to ones that are maximal machines? What happens in the cases where the machine is not maximal?

Associate with the machine the *connection polynomial* $C(x) = x^n - c_1x^{n-1} - \cdots - c_{n-1}x - c_n$, whose coefficients are the connection coefficients. Typically, a cryptography text will say that the condition for the machine to be maximal is that the polynomial $C(x)$ is *primitive* (See, e.g., [7], p. 130, [4], p. 197.) But what does that mean? More generally, what kind of periodicity can occur for an LFSR? This is the question we propose to explore. As a teaser, consider the following question: which of the following integers *cannot* be the period of a 6-cell machine: 6, 7, 8, 9, 10, 11, 12, 21, 30, or 31? The full answer is at the end of the paper. As Beutelspacher wisely suggests ([1], p. 59), it is instructive "to construct – without any theory – your own shift registers of maximum period."

3. Using Linear Algebra

As the name suggests, a Linear Feedback Shift Register can be viewed through the lens of linear algebra. The relation between $x(t+1)$ and $x(t)$ is given by the equation $x(t+1) = Ax(t)$, where A is the matrix

$$A = \begin{pmatrix} c_1 & c_2 & c_3 & \cdots & c_{n-1} & c_n \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{pmatrix}. \tag{3.1}$$

A is non-singular exactly when $c_n \neq 0$; to see that, one can either observe that in that case the *rows* are obviously independent or note that the (mod 2) determinant is c_n. Note that we are doing linear algebra over $\mathbb{F}_p$, the field of integers modulo a prime number p, which is an important tool in cryptography. The matrix is invertible, which implies that the output is periodic for any initial seed. We have $x(t) =$

$A^t x(0)$. The group $GL(n, \mathbb{F}_2)$ of invertible $n \times n$ matrices with entries in $\mathbb{F}_2$ has finite order, and therefore the matrix A has finite order k; that is $A^k = I$, where I is the $n \times n$ identity matrix. So the period of any seed must be a divisor of k. What are the possible values of k?

Using the idea that a matrix is invertible if and only if the columns (or rows) are linearly independent, we can count the number of invertible matrices. This leads to the following well known result:

Proposition 3.1. *The number N of elements of $GL(n, \mathbb{F}_2)$ is given by the formula*

$$\begin{aligned} N &= (2^n - 1)(2^n - 2)(2^n - 2^2) \cdots (2^n - 2^{n-1}) \\ &= 2^{\frac{n^2-n}{2}}(2^n - 1)(2^{n-1} - 1) \cdots (2^2 - 1) \end{aligned}$$

Proof. To construct an invertible matrix, we must choose the n rows(or columns) to be linearly independent vectors. The top row can be anything except the zero vector. The second row can be any other nonzero row, so there are $2^n - 2$ choices. These two vectors span a two-dimensional subspace, so the third row must avoid the four vectors in it, leaving $2^n - 4$ choices. Continuing along these lines, the $k + 1$st row will have $2^n - 2^k$ possible vectors available. □

Since the order of an element of a group divides the order of the group, k must be a divisor of this number. For example, if $n = 6$, then k must divide $N = 2^{15} \times 3^4 \times 5 \times 7^2 \times 31$. In particular $k \neq 11$, giving a partial answer to the teaser. Furthermore, k must be no larger than $2^n - 1$. We will demonstrate this here and again in the next section. The key fact we need is the following algebraic lemma ([3], p. 77).

Lemma 3.2. *Let $P(x) = a_0 + a_1 x + \cdots + x^n$ be a polynomial with coefficients in $\mathbb{F}_p$, p a prime, $a_0 \neq 0$. Then for some $k < p^n$, $P(x)$ is a factor of $x^k - 1$.*

Proof. The ring $R = \mathbb{F}_p[x]/(P(x))$ consists of congruence classes of polynomials with $\mathbb{F}_p$ coefficients, where two polynomials are in the same congruence class if they differ by a multiple of $P(x)$. Using division with remainder, we see that every polynomial is equivalent to exactly one polynomial of degree less than n, and therefore there are exactly $p^n - 1$ non-zero equivalence classes in R. Since there are p^n monomials in the set $\{x^i : 0 \leq i < p^n\}$ there must exist numbers $0 \leq i < j < p^n$ with $x^i \equiv x^j \mod P(x)$. Then $P(x)$ divides $x^j - x^i = x^i(x^{j-i} - 1)$. Since $P(0) \neq 0$, $P(x)$ must divide $x^{j-i} - 1$. □

Now a basic fact from linear algebra is that every $n \times n$ matrix satisfies a polynomial relation of degree n, namely its characteristic equation. In the case of the matrix A, the polynomial is the *connection polynomial* $C(x) = x^n - c_1x^{n-1} - c_2x^{n-2} - \cdots - c_n$, so $C(A) = A^n - c_1A^{n-1} - c_2A^{n-2} - \cdots - c_nI = 0$. Any polynomial $P(x)$ having $C(x)$ as a factor must also satisfy $P(A) = 0$. By Lemma 3.2, $C(x)$ is a factor of $x^k - 1$ for some $k < 2^n$. So $A^k - I = 0$.

So far, we see that the order of A must divide the order of $GL(n, \mathbb{F}_2)$ and be no larger than $2^n - 1$. These conditions, while necessary, are not sufficient. For example, no matrix in $GL(6, \mathbb{F}_2)$ has order exactly 35 or 49. The precise set of conditions is quite complicated; see, for example, [2], pp. 41–43 for the detailed result. In the next section, we indicate a method for finding matrices of given orders.

We conclude this section with an important definition.

Definition 3.3. A polynomial $C(x)$ of degree n in $\mathbb{F}_2[x]$ is *primitive* if it is irreducible and divides $x^{2^n-1} - 1$ but does not divide $x^k - 1$ for $k < 2^n - 1$.

A basic result in the theory of polynomials is that primitive polynomials of degree n exist for all $n \geq 1$. This implies the existence of maximal machines; a machine is maximal if and only if the connection polynomial is primitive.

4. Using Finite Fields

The basic facts we need about finite fields are the following, all of which are elementary:

(1) If p is a prime, then the field $\mathbb{F}_p$ of integers mod p is a finite field with p elements.
(2) If $\mathbb{F}_q$ is a finite field with q elements, then the ring of polynomials $\mathbb{F}_q[x]$ has unique factorization and in fact has a Euclidean algorithm.
(3) If $P(x)$ is an irreducible polynomial of degree n, then the quotient ring $R = \mathbb{F}_p[x]/(P(x))$ is a field with q^n elements. If $P(x)$ is not irreducible, then the resulting quotient ring has zero-divisors.
(4) Every finite field has a subfield $\mathbb{F}_p$ of prime order p; it can be viewed as a vector space over $\mathbb{F}_p$, and therefore has order $q = p^k$ for some prime p and positive integer k.

A less elementary fact is that up to isomorphism there is exactly one field of order p^k. Such a field can be constructed by finding an irreducible polynomial of degree k over $\mathbb{F}_p$ and forming the quotient field. Since there are generally many such polynomials, the uniqueness is certainly not obvious. The standard proof, from the observation that

$\mathbb{F}_q$ is the splitting field of $x^q - x$, is generally not accessible to students in a first cryptography course.

Suppose now that $P(x)$ is irreducible over $\mathbb{F}_p$ of degree n and consider the field $\mathbb{F} = \mathbb{F}_p[x]/(P(x))$. We will always use the symbol α to denote the element of $\mathbb{F}$ representing the congruence class of x. So in $\mathbb{F}$ we have $P(\alpha) = 0$; another way of thinking about this is that $\mathbb{F}$ is obtained by enlarging the field $\mathbb{F}_p$ by throwing in a root of the polynomial $P(x)$.

An element of $\mathbb{F}$ is the congruence class of a polynomial, which can be uniquely taken to be of degree less than n. Therefore the elements are uniquely expressible in the form

$$b = b_{n-1}\alpha^{n-1} + \cdots + b_2\alpha^2 + b_1\alpha + b_0 \qquad b_i \in \mathbb{F}_p$$

Now represent b by the *row* vector $(b_{n-1} \ \ldots \ b_2 \ b_1 \ b_0)$. That is, $\mathbb{F}$ will be viewed as the n-dimensional vector space of row vectors over $\mathbb{F}_p$. Then multiplication by α is a linear map and can be represented by a matrix A; $b\alpha$ is represented by $(b_{n-1} \ \ldots b_2 \ b_1 \ b_0)A$. If the polynomial is the connection polynomial $C(x) = x^n - c_1x^{n-1} - \cdots - c_{n-1}x - c_n$, then the matrix is nothing more than the same matrix A defined earlier in Equation (3.1)!

It follows from this that the order of the matrix A is the order of the element α. It is a fact about finite fields that the multiplicative group of non-zero elements is a cyclic group. A generator of this group is called a primitive element. The polynomial $C(x)$ is primitive if and only if α is a primitive element of the field $\mathbb{F}$. The existence of primitive roots mod p is already an important fact from number theory, and it comes up naturally in a cryptography course. In general α need not be a primitive element, although it is elementary (from group theory) that $\alpha^{p^n-1} = 1$, since the non-zero elements of a field form a group.

Remark 4.1. Once we have found a matrix A as in Equation (3.1) of order $p^n - 1$, we can use it to give a lovely description of the finite field $\mathbb{F}_{p^n}$, namely as the powers of the matrix A together with the zero matrix. See [6] for a discussion of this.

But suppose $C(x)$ is not irreducible? Then the quotient ring R is not a field, and the non-zero elements do not form a group. If we look instead at the *invertible* elements of R, they do form a group. This is the secret key that unlocks the mystery of the LFSR! While the argument below generalizes, let us now return to the special case $p = 2$. In this case addition is the same as subtraction, so we don't have to worry about signs.

The matrix A represents multiplication by α in the ring R. Because $c_n = 1$, we can write

$$1 = (\alpha^{n-1} + c_1\alpha^{n-2} + \cdots + c_{n-1})\alpha$$

Therefore, α is invertible and its order (which is the order of the matrix A) divides the number of invertible elements. This helps explain why the period of an LFSR may be a number not dividing $2^n - 1$. If the polynomial is irreducible, however, α must have order dividing $2^n - 1$.

To determine the possible periods of LFSR's, then, we need to consider the ways in which a polynomial of degree n can be constructed from irreducible polynomials. Here are the basic rules; details may be found in ([3], Chapter 3.) Write $C(x) = g_1g_2 \ldots g_r$, where $g_1, \ldots, g_r$ are pairwise relatively prime. Call the *order* of a polynomial f the order of the element α ; note that this is potentially confusing terminology! Then the order of f is the least common multiple of the orders of g_i. If $g_i = h_i^b$, with h_i irreducible, and the order of h_i is e, then the order of g_i is 2^te, where t is the smallest integer with $2^t \geq b$. Finally, if h has degree k, then its order is a divisor of $2^k - 1$.

5. Example: LFSRs with six cells

Suppose a polynomial $C(x)$ has degree 6. Then since the sum of the degrees of the irreducible factors is 6, $C(x)$ determines a set D of positive integers adding up to 6. We use the notation m^i to represent a factor of degree m repeated i times. For example, $(4, 1^2)$ represents a product of a polynomial of degree 4 and the square of a linear polynomial. The order of an irreducible polynomial of degree 4 must divide $2^4 - 1 = 15$. In fact, such a polynomial will have order 15 or order 5. There is only one linear polynomial, and its square has order 2. Therefore, a polynomial with this pattern must have order 30 or 10. Since $2^2 - 1 = 3$, $2^3 - 1 = 7$, and $2^5 - 1 = 31$ are all primes, irreducible polynomials of degrees 2, 3, and 5 are automatically primitive. The order of a 6th degree polynomial is either 9, 21, or 63. The following table includes all possible cases; each of the 32 possible polynomials fits one of these patterns.

D	$(c_1c_2c_3c_4c_5c_6)$	$Factors$	O
6	(000011)	z^6+z+1	63
6	(010111)	$z^6+z^4+z^2+z+1$	21
6	(001001)	z^3+z+1	9
$5,1$	(101111)	$(z+1)(z^5+z^2+1)$	31
$4,1^2$	(011111)	$(z^4+z+1)(z+1)^2$	30
$4,1^2$	(100011)	$(z^4+z^3+z^2+z^1)(z+1)^2$	10
$4,2$	(111001)	$(z^4+z+1)(z^2+z+1)$	15
$3,1^3$	(001011)	$(1+z)^3(1+z^2+z^3)$	28
$3,2,1$	(010011)	$(1+z)(1+z+z^2)(1+z+z^3)$	21
3^2	(010001)	$(1+z^2+z^3)^2$	14
$3,3$	(111111)	$(z^3+z+1)(z^3+z^2+1)$	7
2^3	(101011)	$(z^2+z+1)^3$	12
$2^2,1^2$	(000001)	$(z+1)^2(z^2+z+1)^2$	6
$2,1^4$	(110111)	$(z+1)^4(z^2+z+1)$	12
1^6	(010101)	$(z+1)^6$	8

So the possible orders of the matrix are 6, 7, 8, 9, 10, 12, 14, 15, 21, 28, 30, 31, and 63. These numbers represent the *largest* periods of seeds in the corresponding machines. It is easy to show that the initial seed $(0,0,0,0,0,1)$ will always achieve the largest period. Of course, the seed $(0,0,0,0,0,0)$ always achieves the shortest period, namely 1. Other periods are also possible, although they must be divisors of the largest period. For example, for the machine with connection polynomial $z^6+z^4+z^3+z^2+z+1$, depending on the initial non-zero seed, the period will be 30, 15, 2, or 1. So perhaps we have not yet uncovered all the secrets of the LFSR.

References

[1] A. Beutelspacher, Cryptology. Translation by J. C. Fisher, Mathematical Association of America, 1994.

[2] S. Golumb, *Shift Register Sequences*, Holden-Day, San Francisco, 1967.

[3] R. Lidl, H. Niederreiter, *Introduction to Finite Fields and Their Applications*, Cambridge Univ. Press, 2000.

[4] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, 1996.

[5] W. Trappe and L. C. Washington, *Introduction to Cryptography with Coding Theory, Second Edition*, Pearson Prentice Hall, Upper Saddle River, N.J., 2006.

[6] W. P. Wardlaw, Matrix representation of finite fields, *Mathematics Magazine*, Vol. **67** (1994), 289–293.

[7] , D. Welsh, *Codes and Cryptography*, Oxford Univ. Press, New York, 1988.