Cryptography and Shift Registers[*]

A.A. Bruen[†] and R.A. Mollin[‡,*]

[†]*Department of Electrical and Computer Engineering, University of Calgary, Canada*

[‡]*Department of Mathematics and Statistics, University of Calgary, Canada*

Abstract: Shift registers are at the heart of cryptography and error-correction. In cryptography they are the main tool for generating long pseudorandom binary sequences which can be used as keys for two communicating parties in symmetric cryptography. Practically all military cryptography makes use of them. Shift registers are also fundamental in signal analysis and *frequency hopping*, most recently in connection with European GPS codes.

The subject has a chequered history. One of the pioneers was a famous Hollywood star who, in her day, was known as the worlds most beautiful woman and wrote one of the earliest patents on frequency hopping which was developed by the US military. We speak of the Austrian-born Hedwig Maria Eva Kiesler, otherwise known as Hedy Lamarr. The development of spread spectrum technology was first proposed by Lamarr using frequency hopping. She obtained a patent after coming to Hollywood in 1942 and turned it over to the US government as a contribution to the war effort. Shortly after the patent expired in 1959 Sylvania digitize the synchronization to supply secure communication during the Cuban missile crisis.

This paper presents a user-friendly, self-contained and comprehensive discussion of the theory of shift registers. Moreover, we provide several examples (and counterexamples) in support of the theory. In the non-linear case we study the relationship between truth tables and de Bruijn sequences. In the linear case, we use a matrix-theoretic approach to describe the situations when the output is periodic and when it is not periodic. Section three describes additional periodicity properties, using the Cayley-Hamilton theorem and the theory of error-correcting codes.

In Section six, we prove a fundamental result to the effect that, for non-singular shift registers of length k, the entire output is uniquely determined by any $2k$ consecutive bits of the output sequence. Although the result is part of the folk lore, it is certainly not well understood and there appears to be a lack of any rigorous proof in the literature. An additional bonus of our proof here is that it can be adapted to provide a new algorithm for demonstrating a little-known, and remarkable fact. Namely, we provide the most general method for constructing two quite different shift registers of the same length that produce identical output sequences! A new result concerning such shift registers is also sketched in Section six.

Keywords: Shift register sequences, continued fractions, periodicity, complexity, coding theory, cryptography,

1. PRELIMINARIES

We may choose to work over any finite field F. Very often, the main case of interest occurs when F is the binary field. In any event, we begin with the concept of a feedback shift register over F.

Let us be given k registers in a row named from left to right

$$R_{k-1}, R_{k-2}, ..., R_1, R_0.$$

Denote the entry in R_i by $x_i \in F$, with $0 \le i \le k-1$. Of course, when F is binary, each of them is either 0 or 1. In the binary case the registers are simply bit-storage registers. An electronic clock controls proceedings. At the first clock pulse the following happens.

Each entry x_i is pushed over one unit to the right to now occupy the register R_{i-1}, for $0 \le i \le k-1$. The right most element x_o gets fed into the *Output* Sequence. Thus, we have a vacancy in the left-most register R_{k-1}. In this register is placed the entry

$$x_n = f(x_{k-1}, x_{k-2}, x_{k-3},...,x_1, x_0)$$

where f is a function of the previous inputs $x_o, x_1, \ldots, x_{k-1}$. Thus the output sequence, from left to right, will be the sequence $\{x_0, x_1, x_2,\ldots,x_{k-1}, f\ldots\}$.

This sequence of elements in the registers represents the current state of the system. The previous state was the sequence $\{x_{k-1}, x_{k-2},\ldots,x_1, x_0\}$. The current state is given by the sequence $\{f, x_{k-1}, x_{k-2},\ldots, x_2, x_1\}$. At each clock pulse we get a new state and a new element for the output sequence. Since there are k registers, we say it has length k.

If the function f is linear, with constant term 0, we say that the feedback shift register Γ is a *linear feedback shift register* (LFSR). In this case, we write

$$f(x_o, x_1, x_2,..., x_{k-2}, x_{k-1}) = c_0x_0+c_1x_1+\cdots+c_{k-2}x_{k-2}+c_{k-1}x_{k-1} \qquad (1.1)$$

*Address correspondence to this author at the Department of Mathematics and Statistics, University of Calgary, Canada:
E-mail: ramollin@math.ucalgary.ca

[*]Mathematics Subject Classification 2000: Primary: 94A55; Secondary: 94A60.

Then we can also put

$$x_k = c_0x_0+c_1x_1 + \cdots + c_{k-2}x_{k-2} + c_{k-1}x_{k-1} \qquad (1.2)$$

Algebraically we can describe Γ as a linear recurrence where the recurrence relation is given by Equation (1.2) above. The (fixed) coefficients $c_0,c_1,c_2,\ldots,c_{k-1}$ are called the *recurrence* or *tap* coefficients of Γ. Thus the recurrence above can also be written as

$$x_{k+i} = c_0x_i + c_1x_{1+i} + c_2x_{2+i} \ldots \qquad (1.3)$$

Initially, in this paper, we will mainly be discussing the linear case. The following examples over the binary field will be useful in the sequel.

Example 1.1. *Let* $k = 3$, *with the initial state given by* $\{x_2, x_1, x_0\} = \{1,1,0\}$. Suppose the recurrence is given by the equation

$$x_3 = x_1 + x_2.$$

Then the successive sequence of states is as follows:

110,011,101,110,...

The sequence of states has fundamental period 3 (*see definitions in section* 2) *as it repeats after* 2 *clock pulses. The output sequence is* {0,1,1,0,1,1,...}. *This sequence also has fundamental period* 3.

Example 1.2. *Again, let* $k = 3$, *with initial state* 001 *and the same recurrence as above. The sequence of states is as follows:* 001,000,000. *The output sequence is* {1,0,0,...}. *Here we have an instance of an output sequence that does not repeat as the leftmost* 1 *never materializes. In other words, the initial state is never repeated.*

Example 1.3. *Let* $k = 3$ *with initial state* 100 *and the same recurrence as above. The sequence of states is given as follows:* 100,110,011,101,110 .... *This sequence is not periodic as the initial state is never repeated. It does form a repeating sequence if we ignore the first state. The output sequence is*

{0,0,1,1,0,1,1,...}.

Example 1.4. *Let* $k = 3$, *with initial state* 011 *and recurrence given by the equation* $x_3 = x_2$. *The sequence of states is* 011,001,000,.... *The output is* {1,1,0,0,0,...} *reminiscent of Example* 1.2; *although the initial states are different. Both terminate with an endless stream of zeros; that is, both go to zero*.

Example 1.5. *Again, let* $k = 3$, *with initial state* {100} *and recurrence defined by the equation* $x_3 = x_0 + x_1$. *The sequence of states is as follows:*

{100,010,101,110,111,011,001,100} ...

The output sequence is {0,0,1,0,1,1,1,...}. *The sequence of states has fundamental period* 7 *as has the output sequence. The total number of possible states is* $2^3 = 8$ *and this number includes the zero state. So there are seven possible non-zero states each of which occurs as a state in the above LFSR.*

2. A MATRIX APPROACH

Let us examine the shift register Γ, with recurrence given by Equation (1.2) above and initial state $u = (x_{k-1}, x_{k-2},\ldots,x_1, x_0)$.

Let $v = (c_0x_0 + \cdots + c_{k-1}x_{k-1},\ldots,x_2, x_1)$

be the state following the clock pulse and

$$A = \begin{pmatrix} c_{k-1} & c_{k-2} & \cdots & c_2 & c_0 \\ 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix} \qquad (2.1)$$

be the *transition matrix*.

Then we can write Equation (1.2) in matrix form as follows.

$$Au^t = v^t \qquad (2.2)$$

We note that the determinant of A equals c_0. It is equal to plus or minus c_0. If $c_0 = 0$ we say that Γ is *singular*. If $c_0 \neq 0$, we say that Γ is *non-singular*. The structure of Γ is determined in large part by whether or not Γ is singular.

Let the output of Γ be given by the sequence $\{x_0, x_1,\ldots, x_{k-1}, x_k,\ldots\}$. We say that this output sequence s has period t for t some positive integer, provided that $x_{i+t} = x_i$, with i =0, 1, 2,.... If t_0 is the smallest integer for which this holds then t_0 is called the *fundamental period* of the output sequence. It is easy to show that t_0 must divide t. Similarly, if $s_0,s_1,s_2,\ldots$ are the successive states of Γ it follows that the state sequence will have the same period t and fundamental period t_0 as the output sequence. Our first fundamental result is as follows.

Theorem 2.1.

1. *If* Γ *is non-singular then, for every initial state, the output sequence is periodic.*
2. *If* Γ *is singular then the output sequence may or may not be periodic.*

Proof. Denote the initial state by u. Assume that the transition matrix A is a non-singular matrix. Then also A^i is non-singular for i = 0, 1.... Our result is clear if u is the zero vector, so assume this is not the case. The total number of non-zero vectors of length k over F is $q^k - 1$, where q is the cardinality of F. Thus the number of states of Γ is at most q^k. It follows that, given u, the vectors in the sequence u, $Au,\ldots, A^iu,\ldots$ cannot all be distinct. Suppose then that $A^su = A^{s+t}u$, with $0 \leq s < s + t \leq q^{k+1}$. Since A is non-singular, A^{-s} exists so that $u = A^tu$. Then, for any $m \geq 0$ we get $A^tA^mu = A^mA^tu = A^mu$. Therefore since each state is of the form A^mu, we see that the state sequence is periodic of period t and part 1 is proved. Part 2 is easily seen by perusing the examples in Section 1.

Remark 2.1. *As mentioned in the proof of Theorem* 2.1, *the maximum number of non-zero states of* Γ *is* $q^k - 1$. *If* Γ *has period* $q^k - 1$ *then we say that* Γ *has maximum period. Such sequences do occur and are extensively used in applications. They have very desirable randomness properties. For example, in the binary case, let us define a block of length j to be a sequence of the form* 0111 ... 10; *i.e. an all ones sequence of length j with zeros at either end. Similarly, a gap of length j is defined by interchanging* 0 *and* 1. *Then it can be shown that, when* $1 \leq j \leq k - 2$, *the total number of blocks and gaps of length j is the same, namely* 2^{k-j-2}.

Theorem 2.2. *Let* Γ *be a given LFSR of length* k*, and transition matrix* A*. If* Γ *is non-singular then the entire output sequence of* Γ *can be determined from any* k *consecutive elements of the output. This is not necessarily true if* Γ *is singular.*

Proof. We think of the given set of k consecutive elements as forming a state s. Then, whether or not Γ is nonsingular, the entire subsequent output of Γ is determined. If Γ is non-singular we can go back, using the inverse of A, to get the previous elements of the output (or we can get them by going forward since the output of Γ is periodic). However, if Γ is singular, then we may not be able to go back. As an illustration, consider Example 1.4. Then modify it by using (010) as the initial state. The two state sequences agree from the second state onwards, yet the outputs of the two shift registers are not the same.

3. A GENERAL STRUCTURE THEOREM

For convenience sake, we assume henceforth that *F has characteristic* 2. In the earlier sections we have seen how the properties of the output sequence of a shift register show much variation and depend very much on the initial state. It appears that not much may be said in general. Remarkably, there is a general result, independent of the initial state, which succinctly describes the structure and output of any non-singular LFSR. This result is best approached through the language of cyclic linear codes.

We use the notation of Section 1, with F being some finite field of characteristic 2. Associated with the recurrence relation (1.2) is the tap polynomial tap(x), where

$$\text{tap}(x)= x^k + c_{k-1}x^{k-1} + \cdots + c_0 \tag{3.1}$$

It is this tap polynomial that glues together the various outputs for our LFSR. We denote it by Γ which is of length k, with transition matrix A given by Equation (2.1). We first need a preliminary result concerning the matrix A.

Theorem 3.1.

1. *The minimal monic polynomial of A is* tap(x).
2. *There exists a positive integer m such that* tap(x) *divides* $x^m - 1$.

Proof. The characteristic polynomial of A is, by calculation, tap(x) which has degree k. By the Cayley Hamilton Theorem, A satisfies tap(x). Let $f(x)$ denote the minimal polynomial of A. If $f(x)$ is unequal tap(x), then $\deg(f(x)) \le k - 1$ and A satisfies $f(x)$. Now the last rows of the matrices A^i for $i = 0,1,2 \ldots .k - 1$ are (00..1), (000010 ...), …, (10000000), which are linearly independent. Thus A satisfies no polynomial equation of degree less than k. This proves part 1.

To prove part 2, note that the $k\times k$ matrices $A, A^2, A^3, \ldots.$ over the finite field F cannot all be distinct. Thus, for some $i<j$ we have $A^i = A^j$, so that $A^{j-i} = 1$. Since the minimal polynomial of A is tap(x), we see that tap(x) divides $x^m - 1$ where $m = j - i$. This proves part 2.

Let n be any positive integer. Then, a code C of length n over F is any non-empty set of n-tuples over F. These n-tuples in C are called the *code words* of C. If the set of code words is closed under addition and scalar multiplication then C is linear. In this case we can pick a basis for C consisting of m vectors say where m is the dimension of C. We then form a generator matrix G for the code C of size $m \times n$ and of rank m, where $m=\dim(C)$.

Each vector $c=(c_0, c_1,\ldots,c_n)$ of the linear code C is associated with the polynomial

$$c(x) = c_0 + c_1x + \ldots + c_{n-1}x^{n-1}. \tag{3.2}$$

The linear code C is said to be cyclic if whenever $c \in C$ then the code vector c_1 is in C, where

$$c_1 = (c_n, c_0, c_1, c_2, \ldots, c_{n-1}),$$

is obtained from c by a cyclic shift of order one, namely a single shift to the right. Algebraically this shift corresponds to multiplying the polynomial of c by x with the proviso that $x^n = 1$. A shift of order u to the right corresponds to multiplication by x^u. Since C is invariant under all shifts we see that, algebraically, C corresponds exactly to an ideal J in the polynomial ring $F[x]/x^{n-1}$. Denote the unique monic polynomial $g(x)$ of smallest degree in J by $g(x)$. It will follow that $g(x)$ must divide $x^n - 1$. Also, $g(x)$ generates J. *This means that all polynomials in J* are obtained by multiplying g by a polynomial and reducing modulo x^{n-1}. A generator matrix G for the code C is provided by the matrix whose rows are the vectors corresponding to the polynomials $g(x)$, $xg(x)$, $x^2g(x)$, ..., $x^{\{n-1-k\}}g(x)$, where $k = \deg(g)$. Thus the dimension of C is $n - k$; with $n \ge k$. Each vector in this basis for C is obtained from the previous vector in the basis by a cyclic shift to the right.

Associated with each cyclic linear linear code C is another such code $C^\perp$ consisting of all vectors perpendicular to all vectors in C, namely those consisting of all vectors of length n whose dot product with every vector in C is zero. From elementary linear algebra the dimension of $C^\perp$ is equal to

$$n - \dim C = n - (n - k) = k = \deg(g). \tag{3.3}$$

Returning to our shift registers of length k, let n denote the smallest positive integer such that $tap(x)$ divides $x^n - 1$. Recalling that the monic polynomial $tap(x)$ has degree k, we can now form the cyclic linear code C of length n and dimension $n - k$ with generator polynomial $tap(x)= g(x)$. Recall that for any polynomial $f(x) = c_0 + c_1x + \cdots + c_tx^t$ with $c_t = 0$, of degree k, the revers*e* polynomia*l* $f_1(x)$ is given by

$$f_1(x) = c_t + c_{t-1}x + \cdots + c_0x^t.$$

Note that if f has non-zero constant term then both f and f_1 have degree t. We have $x^n - 1= g(x)h(x)$ where $\deg(h(x)) = n - \deg(g(x)) = n - k$.

The following result is standard.

Theorem 3.2. *The code* $C^\perp$ *is generated by* $h_1(x)$*, the reverse polynomial of* $h(x)$.

Proof. The vanishing of the dot products of the code word g and all its shifts with h_1 and all its shifts is equivalent to the vanishing of the various coefficients of $g(x)h(x) = x^n - 1$. Since $h_1(x)$ has degree $n - k$, its right cyclic shifts-including the null shift-give a cyclic code of dimension k, and we are done.

Using the notation above, with Γ assumed to be non-singular, we can now prove our main result.

Theorem 3.3.

1. *For any non-zero initial state, the output sequence of the non-singular LFSR* Γ *has fundamental period dividing n where n is the smallest positive integer for which the tap polynomial divides the polynomial* $x^n - 1$ *in the polynomial ring* $F[x]$.

2. *Every possible output sequence of length n is a vector in* $C^{\perp}$*, and vice versa.*

3. *There exists at least one output sequence with fundamental period equal to n.*

Proof. Consider an output sequence $x = (x_0, x_1, x_2, \ldots, x_{n-1})$ of length n . The fact that x is an output sequence implies that x is perpendicular to the vector corresponding to $g(x)$ = tap(x) given by

$$c = (c_0, c_1, c_2, \ldots, c_{k-1}, 1, 0, \ldots, 0)$$

and its $n-k$ cyclic shifts to the right, including the null shift. Thus, x is in $C^{\perp}$ and vice versa. This shows part 2.

To show part 1, we have that x, being in $C^{\perp}$, is perpendicular, not just to the first $n-k$ shifts of c but to all n shifts of c. But this is equivalent to x having period n.

Finally, we come to part 3. We examine the codeword x corresponding to $h_1(x)$. Suppose that x has period u with $u < n$. Then u must divide n, and the elements of x consists of n/u identical blocks of u symbols. The same must hold for the vector corresponding to h_1. Then, algebraically, we have

Therefore,

$$h_1(x)[x^u - 1] = w(x)[x^n - 1].$$

Multiplying by $g(x)$, using the fact that the $g(x)h(x) = x^n - 1$, and canceling the term $x^n - 1$, we get that

$$g(x)w(x) = x^u - 1$$

contradicting the fact that n is the smallest exponent for which $g(x)$ divides $x^n - 1$.

4. WHEN THE TAP POLYNOMIAL IS IRREDUCIBLE

We use the notation of previous sections. We are given a non-singular LFSR, denoted by Γ, with transition matrix A. The period of the matrix A is the smallest positive integer t such that $A^t = I$ [see Equation (3.1)]. The field F has characteristic 2. Our main result is as follows.

Theorem 4.1. *Assume that* tap(x) *is an irreducible polynomial in* $F[x]$. *Then*

1. *For any non-zero initial state, the output sequence has fundamental period t where t is the period of A.*

2. *t divides* $q^k - 1$.

Proof. We can form the matrix ring over A modulo tap(x). Since tap(x) is irreducible this ring is a field, so all elements are invertible. Suppose that for some non-zero state x we have $A^m x = x$ with $m < t$. Then $(A^m - I)x = 0$. Since $A^m - I$ is either zero or invertible, we conclude that $A^m = I$. But this contradicts the minimality of t, proving part 1.

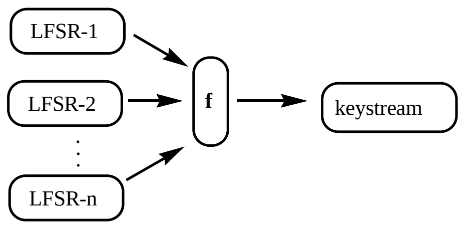
For each non-zero state as input, the resulting state has period t. Thus, since the total number of non-zero vectors of length k over F is $q^k - 1$, part 2 follows.

Theorem 4.2. *A non-singular LFSR has maximum period if and only if* tap(x) *is irreducible and does not divide* $x^n - 1$ *for any* $n < q^k - 1$, *where q is the cardinality of the field and k is the length of the LFSR.*

Proof. Most of this can be seen from our previous work. The remaining difficulty relies on showing that a maximum period implies that tap(x) is irreducible. The details of this are to be found in [1].

5. NON-LINEAR SHIFT REGISTERS

LFSRs are notoriously insecure from a cryptographic viewpoint. The reason is that all a cryptanalyst needs is a few bits of consecutive plaintext and corresponding cipher text to determine the key. Bitstrings output by a single LFSR are not secure since sequential bits are linear, so it only takes 2k output bits (where k is the length of the LFSR) to determine it. However, one can use LFSRs as building blocks for more secure systems. One mechanism is to use several of them in parallel, meaning that $n > 1$ LFSRs are input to a function f, called the *combining function*, which outputs a keystream. Such a system is called a *nonlinear combination generator*, illustrated below



There exists a method of nonlinear combination generation called *clock controlled generation* that attempts to foil attacks based upon the regular the clocking (or stepping) of the other LFSRs. Necessarily, LFSRs become a trade-off between speed and security If one is not concerned with security but rather speed, such as in cable television transmission, then LFSRs are a good bet. For systems of communications requiring high security, they are not. However, they can be built into *non-*linear pseudo-random number generators.

We conclude with some general remarks about nonlinear shift registers. First, it is important to note that a nonlinear recursion can be described using its truth table. For instance, we study the following situation.

If k is the length of the LFSR, there exist maximal sequences of period length 2^k, which are called *de Bruijn* sequences. This is illustrated as follows.

x_0	x_1	x_2	$f(x_0,x_1,x_2)$
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Here

$$f(x_0,x_1,x_2) = 1+ x_0 + x_1 + x_1x_2.$$

There are few results on finding the general period length of a nonlinear shift register when such exists. This seems to be a difficult problem.

6. DETERMINING THE OUTPUT FROM 2*k* bits

The result in this section says that, under certain conditions (that are both subtle and general) the entire output of a shift register of length k can be determined from any $2k$ consecutive bits under a nonsingularity assumption. (The proof has the useful application of providing a method for constructing two different shift registers of the same length having the same output sequence.) Although the result is officially "well-known" in the "folk-lore" of the subject we have not seen a formal proof; moreover, non-singularity is crucial.

Theorem 6.1. *Let Γ_k be a non-singular linear shift register of length k. Then the entire output of Σ_k can be determined from any $2k$ consecutive bits of the sequence, provided that* $c_0 = 1$.

Proof. Suppose the initial state vector is $\{x_{k-1}, x_{k-2},\ldots,x_0\}$ with recurrence relation $x_k = \sum_{i=0}^{k=1} c_i x_i$. Assume that $2k$ consecutive output bits are known and denoted by $\alpha_0, \alpha_1,\ldots,\alpha_{2k-1}$. Then the recurrence relation for Γ_k is expressible in the following matrix equation. If we set

$$P = \begin{pmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \cdots & \alpha_{k-1} \\ \alpha_1 & \alpha_2 & \alpha_3 & \cdots & \alpha_k \\ \alpha_2 & \alpha_3 & \alpha_4 & \cdots & \alpha_{k-1} \\ \vdots & \vdots & \vdots & & \vdots \\ \alpha_{k-1} & \alpha_k & \alpha_{k-1} & \cdots & \alpha_{2k-2} \end{pmatrix}$$

$$C = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{k-2} \\ c_{k-1} \end{pmatrix} \quad \text{and} \quad Z = \begin{pmatrix} \alpha_k \\ \alpha_{k-1} \\ \vdots \\ \alpha_{2k-2} \\ \alpha_{2k-1} \end{pmatrix}$$

then

$$PC = Z. \tag{6.1}$$

If P is invertible then multiplying by P^{-1} will yield the recurrence coefficients of Γ_k, from which the entire sequence can be determined, and the result is established.

If P is not invertible, then the equation $PX = Z$ will have more than one solution $X = C$. If $D \neq C$ is such a solution, then $PD = Z$ so $P(C + D)= PC + PD = Z + Z$ is the zero matrix since the addition is in a field of characteristic 2. By taking transposes, we get that $(C + D)^tP^t$ is also the zero matrix where $(C + D)^t$ is a nonzero row vector and $P = P^t$ since P is a symmetric matrix.

We now wish to show that the entire output sequence for C equals that of D given that they agree on some consecutive $2k$ output bits. To do this we first show that the two output sequences will also agree on the succeeding entry if and only if equation (6.1), shifted by one bit holds, in the fashion explained below.

Let

$$P_1 = \begin{pmatrix} \alpha_1 & \alpha_2 & \alpha_3 & \cdots & \alpha_k \\ \alpha_2 & \alpha_3 & \alpha_4 & \cdots & \alpha_{k-1} \\ \vdots & \vdots & \vdots & & \vdots \\ \alpha_k & \alpha_{k+1} & \alpha_{k+2} & \cdots & \alpha_{2k-1} \end{pmatrix}$$

We need to prove that $P_1C = P_1D$ where by assumption,

$$P_1C = \begin{pmatrix} \alpha_{k+1} \\ \alpha_{k+2} \\ \vdots \\ \alpha_{2k-1} \\ \alpha_{2k} \end{pmatrix}.$$

If we set $U =(C + D)^t$, then we know from above that UP is the zero matrix, so $U(PC) = (UP)C$ is as well. Thus, the matrix product of U with each column of P and also with the right column Z is zero. Each column of P_1 apart from the last, is a column of P. Also, Z is the same as the last column of P_1. Hence, UP_1 is the zero matrix. Transposing, we get $P_1^t(C+D)$ is the zero matrix, so that $P_1(C+D)$ is the zero matrix since P_1 is symmetric. This implies that $P_1C = P_1D$ as we needed.

Thus far, we have shown that if two sequences agree on any $2k$ consecutive bits, then they agree on all subsequent bits. Now it remains to show that if they agree on $2k$ consecutive bits, they agree on all *preceding* bits.

We assume that

$$P_1C = P_1D = \begin{pmatrix} \alpha_{k+1} \\ \alpha_{k+2} \\ \vdots \\ \alpha_{2k-1} \\ \alpha_{2k} \end{pmatrix}$$

and we show that $PC = PD$ where

$$PC = \begin{pmatrix} \alpha_k \\ \alpha_{k+1} \\ \vdots \\ \alpha_{2k-2} \\ \alpha_{2k-1} \end{pmatrix}.$$

As in the above argument UP_1 is the zero matrix. Therefore, $UP = (\lambda, 0, 0, \ldots, 0)$ since the matrix product of U with all columns of P, except possibly the first, is zero. Also, the matrix product of U with the last column of P_1 is zero, and this column is PC, so we have that $U(PC)$ is the zero matrix. By associativity,

$$(UP)C = (\lambda, 0, 0, ..., 0)C$$

so $\lambda c_0 = 0$ giving $\lambda = 0$ since $c_0 \neq 0$. Thus, UP is the zero matrix, so $(C + D)^t P$ is the zero matrix, and by transposing, $P^t(C + D)$ is the zero matrix so $P(C + D)$ must be as well. This means that $PC = PD$, so C and D agree on all preceding bits, thereby completing the proof.

Theorem 6.2. *Let Γ_1 be a LFSR of length k and let Γ_2 be a LFSR of the same length. Suppose that the outputs of Γ_1, Γ_2 agree on 2k consecutive elements. Then*

1. *If at least one of Γ_1, Γ_2 is non-singular then Γ_1 and Γ_2 have identical output sequences*
2. *If they are both singular this need not be the case.*

Proof. If both are non-singular they are both periodic by part 1 of Theorem 2.1. In this case the result follows from the mere fact, as shown in Theorem 6.1, that since the two output sequences agree on $2k$ bits they agree on all subsequent bits. If at least one is non-singular the preceding entries will also be equal as shown in Theorem 6.1.

For part 2, we examine Examples 1.2, 1.4. Both outputs have the six successive elements 000000 in common but their outputs disagree in the second position.

Remark 6.1. *It is easy to construct examples showing that the restriction to consecutive elements is absolutely necessary in Theorem* 6.2. *The proof of Theorem* 6.2 *suggests the intriguing possibility that two different LFSRs produce the same output! This is indeed the case. Our next result clarifies the situation.*

Theorem 6.3. *It is possible for two different LFSR of length k to produce the same output sequence. In such a case this common output sequence can be produced by another LFSR of shorter length.*

Sketch of proof. Using the notation of Theorem 6.1 we have that $UP = UP_1 = UP_2 \ldots$ Thus U causes a dependency among the rows of P, P_1, P_2, $P_3 \ldots$ This in turn yields an LFSR of length less than k that produces the given output sequence. For an example, let $k = 4$ and let Γ_1, Γ_2 have initial state x_3, x_2, x_1, x_0 = 1,0,1,1. Let Γ_1 and Γ_2 have recurrence given by $x_4 = x_0 + x_2$, $x_4 = x_0 + x_1 + x_3$, respectively. The common output sequence has period 3 and is 110110110 … This same output can be generated by the length 2 LFSR with initial state 11 and the Fibonacci recurrence given by $x_2 = x_1 + x_0$.

Corollary 6.1. *The shortest LFSR that generates a given sequence is unique.*

Remark 6.2. *One can use the Berlakamp-Massey algorithm to determine the shortest LFSR given in Corollary* 6.1.

Note that the reader may find more information on the background to the above discussion in [2, 3].

ACKNOWLEDGEMENT

Both authors gratefully acknowledge the support of NSERC Canada grants #A8726 and # A8484, respectively.

REFERENCES

[1] Ash RB. Information theory. Mineola, USA: Dover 1990.
[2] Bruen AA, Forcinito MA. Cryptography, Information theory, and error-correction. Hoboken, USA: Wiley 2005.
[3] Mollin RA. An introduction to cryptography. 2nd ed. Boca Raton, USA: Chapman and Hall/CRC, Taylor and Francis Group 2007.

Received: January 09, 2008 Revised: March 24, 2009 Accepted: June 15, 2009