A List of Maximum Period NLFSRs

Elena Dubrova

Royal Institute of Technology (KTH), Forum 120, 164 40 Kista, Sweden
{dubrova}@kth.se

Abstract. Non-Linear Feedback Shift Registers (NLFSRs) are a generalization of Linear Feedback Shift Registers (LFSRs) in which a current state is a non-linear function of the previous state. While the theory behind LFSRs is well-understood, many fundamental problems related to NLFSRs remain open. Probably the most important one is finding a systematic procedure for constructing NLFSRs with a guaranteed long period. Available algorithms either consider some special cases, or are applicable to small NLFSRs only. In this paper, we present a complete list of n-bit NLFSRs with the period $2^n - 1$, $n < 25$, for three different types of feedback functions with algebraic degree two. We hope that the presented experimental data might help analysing feedback functions of maximum-period NLFSRs and finding a supporting theory characterizing them.

1 Introduction

Linear Feedback Shift Registers (LFSRs) are one of the most popular devices for generating pseudo-random sequences. They have numerous applications, including error detection and correction [29], data compression [30], testing [28], and cryptography [34]. The research on LFSRs started in the early 60s [18] and continued actively for many years. Today LFSRs are a mature and well-understood subject. Most fundamental problems related to LFSRs are solved. It is known how to synthesise an LFSR with a maximum period - one has to use a primitive generator polynomial[1]. A minimal LFSR generating a given binary sequence can be constructed using Berlekamp-Massey algorithm [27]. Statistical properties of sequences generated by LFSRs have been characterized by Golomb's postulates [18].

Non-Linear Feedback Shift Registers (NLFSR) are a generalization of LFSRs in which a current state is a non-linear function of the previous state [21]. While the theory behind LFSRs is well-understood, many fundamental problems related to NLFSRs remain open. Probably the most important one is finding a systematic procedure for constructing NLFSRs with a guaranteed long period. Available algorithms either consider some special cases [10], or are applicable to small NLFSRs only [1, 3, 4, 11–13, 15–17, 19–21, 23, 31–33] (see [14] for an excellent overview). The general problem is hard because there seems to be no simple algebraic theory supporting it.

In this paper, we present a complete list of n-bit NLFSRs with the period $2^n - 1$, $n < 25$, for the following types of feedback functions with algebraic degree[2] two:

[1] An irreducible polynomial of degree n is called *primitive* if the smallest m for which it divides $x^m + 1$ is equal to $2^n - 1$ [24].

[2] The *algebraic degree* of a Boolean function is the number of variables in the largest product-term of its algebraic normal form [5].

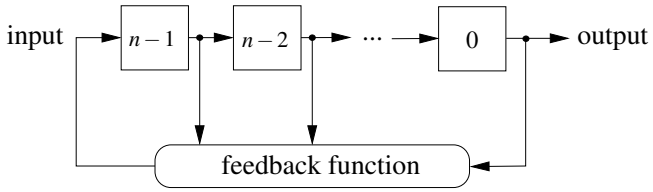

Fig. 1. The general structure of an n-bit Fibonacci FSR.

1. $f(x_0, x_1, \ldots, x_{n-1}) = x_0 \oplus x_a \oplus x_b \oplus x_c \cdot x_d$
2. $f(x_0, x_1, \ldots, x_{n-1}) = x_0 \oplus x_a \oplus x_b \cdot x_c \oplus x_d \cdot x_e$
3. $f(x_0, x_1, \ldots, x_{n-1}) = x_0 \oplus x_a \oplus x_b \oplus x_c \oplus x_d \oplus x_e \cdot x_h$

where $a, b, c, d, e, h \in \{1, 2, \ldots, n-1\}$, "$\oplus$" is the XOR (addition modulo 2) and "$\cdot$" is the AND (multiplication modulo 2).

The maximum possible period for an n-bit NLFSR is 2^n. The NLFSRs presented in Section 4 do not include all-0 state in their longest cycle of states. They can be extended to have the period 2^n by adding to their feedback functions a product-term $\bar{x}_1\bar{x}_2 \ldots \bar{x}_n$, where $\bar{x}$ denotes the complement of x defined by $\bar{x} = x \oplus 1$. One can see that this increases the circuit complexity of feedback functions, which is undesirable for many applications. For this reason, we focus on NLFSRs with the period $2^n - 1$.

We hope that the presented experimental data might help analysing feedback functions of maximum-period NLFSRs and finding a supporting theory characterizing them.

The paper is organized as follows. Section 2 gives a background on feedback shift registers. Section 3 describes important properties of NLFSRs which help reducing the search space for maximum-period NLFSRs. Section 4 presents a list of feedback functions of NLFSRs with the period $2^n - 1$. Section 5 concludes the paper.

2 Background

A *Feedback Shift Register (FSR)* consists of n binary storage elements, called *stages* or *bits*. Each stage $i \in \{0, 1, \ldots, n-1\}$ has an associated *state variable* x_i which represents the current value of the stage i and a *feedback function* $f_i : \{0,1\}^n \to \{0,1\}$ which determines how the value of i is updated.

A *state* of an FSR is a vector of values of its state variables $(x_0 x_1 \ldots x_{n-1})$. At every clock cycle, the next state of an FSR is determined from its current state by simultaneously updating the value of each stage i to the value of f_i. The *period* of an FSR is the length of the longest cyclic output sequence it produces. The *output* of an FSR is the value of its stage 0. The *input* of an FSR is the value of its stage $n-1$.

If all feedback functions of an FSR are linear, i.e. of type $f(x_0, x_1, \ldots, x_{n-1}) = c_0 \oplus c_1 x_0 \oplus c_2 x_1 \oplus \ldots c_n x_{n-1}$, where $c_i \in \{0,1\}$ for $i \in \{0, 1, \ldots, n\}$, then it is called a *Linear Feedback Shift Register* (LFSR). Otherwise, it is called a *Non-Linear Feedback Shift Register* (NLFSR).

An FSR can be implemented either in the *Fibonacci* or in the *Galois* configuration. In the former, the feedback is applied to the input stage of the shift register only

(Figure 1), while in the latter the feedback can potentially be applied to every stage. Figure 2 shows an example of a 4-bit Fibonacci NLFSR with the feedback function $f(x_0,x_1,x_2,x_3) = x_0 \oplus x_1 \oplus x_2 \oplus x_1x_2$.

For LFSRs, the exist a unique transformation between the Galois and the Fibonacci configurations. The Galois configuration can be obtained from the Fibonacci one (and vice verse) by reversing the order of LFSR's feedback taps and adjusting the initial state. For NLFSRs, the Fibonacci to Galois transformation is not unique [2,8,9]. In this paper, we present results for the Fibonacci NLFSRs only. One can convert the NLFSRs presented in Section 4 to the Galois configuration by applying the transformation from [8].

The first algorithm for constructing a minimal NLFSR generating a given binary sequence was presented by Jansen et. al. in [22]. Alternative algorithms were given by Linardatos et. al. [26] and Limniotis et. al. [25].

3 Properties of NLFSRs

In this section, we describe important properties of NLFSRs which help reducing search space for maximum-period NLFSRs.

The state transition graph of an n-bit Fibonacci NLFSR is *branchless*, i.e. consists of pure cycles, if and only if its feedback function is of type

$$f(x_0,x_1,\ldots,x_{n-1}) = x_0 \oplus g(x_1,x_2,...,x_{n-1}) \tag{1}$$

where the function g does not depend on the variable x_0 [18]. This is because, if f is of type (1), then the states $(x_0x_1\ldots x_{n-1})$ and $(\bar{x}_0x_1\ldots x_{n-1})$ each have a different successor depending on the value of g. The value of g is the same for the states $(x_0x_1\ldots x_{n-1})$ and $(\bar{x}_0x_1\ldots x_{n-1})$, while the values of x_0 and $\bar{x}_0$ are different. On the other hand, if both states $(x_0x_1\ldots x_{n-1})$ and $(\bar{x}_0x_1\ldots x_{n-1})$ have the same successor, then f cannot be of type (1).

There are $2^{2^{n-1}-n+1}$ different n-bit Fibonacci NLFSRs with the period 2^n-1 [14]. This formula can be derived as follows. Consider a graph G_n which has 2^n nodes, representing all possible states of an n-bit NLFSR, and 2^{n+1} edges, representing all possible transition between these states. Each node of G_n has two possible predecessors and two possible successors.

We can find all cycles of length 2^n in G_n by finding all possible Hamiltonian paths[3]. De Bruijn [6] has shown that is the graph G_n has k Hamiltonian paths, then the graph G_{n+1} has $2^{2^{n-1}-1} \cdot k$ Hamiltonian paths. Since the graph G_1 has one Hamiltonian path, by induction, the number of Hamiltonian paths in G_n is given by $2^{2^{n-1}-n}$.

To form a cycle of length 2^n-1 from a cycle of length 2^n, we can remove the loop at node $00\ldots0$ or the loop at node $11\ldots1$ of G_n. Since there are no other loops in G_n, there are exactly two cycles of length 2^n-1 for each cycle of length 2^n. Therefore, the number of cycles of length 2^n-1 in G_n is $2 \cdot 2^{2^{n-1}-n}$.

[3] A *Hamiltonian path* in a graph G is a path which goes through all of the vertices of G once and only once [7].

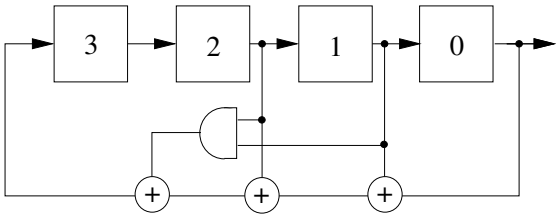

Fig. 2. An example of a 4-bit Fibonacci NLFSR.

The set of n-bit Fibonacci NLFSRs with the period $2^n - 1$ can be partitioned into 4 subsets [21]: (1) basic, (2) reverse of basic, (3) complement of basic, and (4) reverse complement of basic, each of size $2^{2^{n-1}-n-1}$. If the basic NLFSR has the feedback function of type (1), then reverse, complement, and reverse complement NLFSRs have the following feedback functions:

$$
\begin{array}{ll}
f_r(x_0,x_1,\ldots,x_{n-1}) = x_0 \oplus g(x_{n-1},x_{n-2},...,x_1) & \text{reverse} \\
f_c(x_0,x_1,\ldots,x_{n-1}) = x_0 \oplus 1 \oplus g(x_1,x_2,...,x_{n-1}) & \text{complement} \\
f_{rc}(x_0,x_1,\ldots,x_{n-1}) = x_0 \oplus 1 \oplus g(x_{n-1},x_{n-2},...,x_1) & \text{reverse complement}
\end{array}
$$

These NLFSRs generate sequences which are reverse, complement, or reverse complement of the basic sequence, correspondingly. In the list presented in the next section, we show only the basic case. Other cases are not included.

4 List of feedback functions of n-bit Fibonacci NLFSRs with the period $2^n - 1$

In the format we use in the list below, indexes of variables of each product-term of a feedback function are separated by a comma. If a product-term contains more than one variable, we put round brackets around the indexes of its variables. For example, the function $f(x_0,x_1,x_2,x_3) = x_0 \oplus x_1 \oplus x_2 \oplus x_1x_2$ is represented as 0,1,2,(1,2).

n = 4:
0,1,2,(1,2)
0,1,2,(1,3)
0,1,2,(2,3)
0,1,(1,2),(2,3)
0,2,(1,2),(1,3)

n = 5:	**n = 6:**	**n = 7:**	**n = 8:**	**n = 9:**
0,1,2,(2,4)	0,1,2,(1,2)	0,1,2,(2,6)	0,1,5,(1,5)	0,1,6,(4,6)
0,1,3,(1,3)	0,1,2,(2,4)	0,1,4,(1,3)	0,1,6,(1,2)	0,1,6,(4,8)
0,1,3,(1,4)	0,1,3,(1,5)	0,1,5,(1,5)	0,1,6,(1,7)	0,2,4,(4,5)
0,1,3,(2,3)	0,1,4,(1,4)	0,1,5,(3,5)	0,1,6,(2,4)	0,3,4,(3,7)
0,1,(1,2),(2,3)	0,2,3,(1,3)	0,1,5,(4,6)	0,1,6,(4,5)	0,1,(1,5),(2,5)
0,1,(1,2),(3,4)	0,2,3,(1,5)	0,2,4,(1,2)	0,1,6,(5,6)	0,1,(1,6),(6,7)
0,1,(1,3),(2,3)	0,2,3,(2,3)	0,2,4,(2,5)	0,2,5,(2,4)	0,1,(1,8),(2,7)
0,1,(1,3),(2,4)	0,2,3,(2,4)	0,1,(1,2),(5,6)	0,2,5,(3,7)	0,1,(1,8),(5,6)
0,1,(1,4),(2,3)	0,1,(1,2),(4,5)	0,1,(1,5),(3,4)	0,2,5,(4,5)	0,1,(2,3),(3,8)
0,2,(1,2),(3,4)	0,1,(1,3),(3,5)	0,1,(1,6),(4,5)	0,3,4,(2,4)	0,1,(2,8),(3,7)
0,2,(1,3),(2,4)	0,1,(2,3),(2,5)	0,1,(2,3),(3,5)	0,3,4,(2,7)	0,1,(3,4),(3,5)
0,2,(1,4),(2,3)	0,2,(1,3),(2,4)	0,1,(2,5),(3,5)	0,3,4,(3,4)	0,1,(3,7),(5,8)
0,2,(1,4),(2,4)	0,2,(1,3),(3,4)	0,1,(2,5),(4,5)	0,3,4,(4,6)	0,2,(1,5),(4,6)
	0,2,(1,3),(3,5)	0,1,(3,4),(4,5)	0,3,4,(4,7)	0,2,(1,6),(2,7)
	0,2,(1,5),(2,4)	0,2,(1,2),(4,6)	0,3,4,(6,7)	0,2,(1,8),(3,4)
	0,2,(1,5),(4,5)	0,2,(1,4),(3,4)	0,1,(1,4),(2,4)	0,2,(2,7),(4,6)
	0,2,(2,3),(3,5)	0,2,(1,5),(2,6)	0,1,(1,6),(2,5)	0,2,(4,7),(5,6)
	0,2,(3,4),(3,5)	0,2,(1,6),(2,4)	0,1,(2,3),(2,4)	0,3,(1,2),(4,7)
	0,3,(1,4),(2,3)	0,2,(1,6),(3,6)	0,1,(2,4),(6,7)	0,3,(1,6),(1,7)
	0,3,(1,4),(2,4)	0,2,(1,6),(5,6)	0,1,(3,4),(4,7)	0,3,(1,7),(4,8)
	0,3,(1,4),(3,4)	0,2,(2,4),(3,5)	0,2,(1,3),(4,6)	0,3,(2,3),(4,7)
		0,2,(2,5),(4,6)	0,2,(1,3),(5,7)	0,4,(1,3),(2,8)
		0,2,(2,6),(4,6)	0,2,(1,5),(6,7)	0,4,(1,6),(3,6)
		0,2,(3,6),(5,6)	0,2,(1,7),(2,3)	0,4,(2,3),(5,8)
		0,3,(1,2),(2,3)	0,2,(3,7),(6,7)	0,4,(2,5),(2,8)
		0,3,(1,3),(1,6)	0,3,(1,2),(2,4)	0,4,(2,7),(3,8)
		0,3,(1,4),(3,6)	0,3,(1,4),(2,4)	0,4,(2,8),(6,7)
		0,3,(1,5),(3,5)	0,3,(1,6),(3,6)	0,4,(3,5),(3,7)
		0,3,(1,6),(3,4)	0,3,(1,6),(4,6)	0,1,2,3,4,(3,7)
		0,3,(2,3),(4,5)	0,3,(1,6),(4,7)	0,1,2,3,7,(4,6)
		0,3,(2,5),(3,5)	0,3,(2,3),(5,6)	0,1,2,4,7,(1,6)
		0,1,2,3,4,(1,6)	0,3,(2,4),(6,7)	0,1,2,5,6,(1,6)
		0,1,2,3,4,(2,3)	0,3,(2,6),(3,7)	0,1,2,5,6,(2,6)
		0,1,2,3,4,(2,6)	0,1,2,3,5,(2,6)	0,1,2,5,8,(2,6)
		0,1,2,3,6,(1,3)	0,1,2,3,6,(3,5)	0,1,2,6,7,(3,6)
		0,1,2,3,6,(1,5)	0,1,2,3,6,(5,7)	0,1,3,4,5,(3,7)
		0,1,2,3,6,(2,6)	0,1,2,4,5,(2,4)	0,1,3,5,7,(5,6)
		0,1,2,4,5,(1,2)	0,1,2,4,7,(1,5)	0,1,3,5,8,(3,5)
		0,1,2,4,5,(1,5)	0,1,2,5,7,(2,4)	0,1,4,6,7,(1,7)
		0,1,2,4,5,(2,6)	0,1,3,4,7,(1,4)	0,2,3,4,7,(2,8)
			0,1,3,4,7,(1,6)	
			0,1,3,4,7,(3,7)	

n = 10:
0,1,2,(8,9)
0,1,4,(3,7)
0,1,8,(6,7)
0,2,5,(1,5)
0,4,5,(2,6)
0,4,5,(4,8)
0,4,5,(4,9)
0,1,(1,2),(3,4)
0,1,(2,4),(2,5)
0,1,(2,8),(7,9)
0,1,(3,8),(4,7)
0,1,(4,8),(6,7)
0,2,(1,3),(4,7)
0,2,(1,4),(3,7)
0,2,(1,5),(3,5)
0,2,(1,5),(4,9)
0,2,(1,6),(1,7)
0,2,(1,7),(4,6)
0,2,(1,9),(5,9)
0,2,(3,5),(3,7)
0,2,(3,9),(8,9)
0,3,(1,2),(2,8)
0,3,(1,3),(7,9)
0,3,(1,6),(3,8)
0,3,(1,6),(6,9)
0,3,(2,3),(2,6)
0,3,(2,7),(8,9)
0,3,(2,8),(7,9)
0,3,(6,7),(8,9)
0,4,(1,3),(1,7)
0,4,(1,3),(7,8)
0,4,(1,3),(7,9)
0,4,(1,5),(1,9)
0,4,(1,5),(7,9)
0,4,(7,8),(7,9)
0,1,2,4,8,(1,5)
0,1,2,4,8,(2,4)
0,1,2,5,8,(5,9)
0,1,3,4,7,(3,6)
0,1,3,6,7,(1,6)
0,1,4,5,9,(1,9)
0,1,4,5,9,(4,9)
0,1,4,5,9,(5,9)
0,1,5,6,7,(2,8)
0,2,3,4,6,(3,6)
0,2,4,5,8,(2,4)
0,2,4,6,7,(1,6)

n = 11:
0,1,9,(1,4)
0,2,5,(1,9)
0,2,8,(6,9)
0,1,(1,7),(2,8)
0,1,(1,9),(2,7)
0,1,(2,3),(4,5)
0,1,(2,5),(3,4)
0,1,(2,7),(3,10)
0,1,(3,7),(3,8)
0,1,(3,7),(7,8)
0,2,(4,5),(6,10)
0,2,(4,6),(9,10)
0,2,(7,9),(8,10)
0,3,(1,6),(8,9)
0,3,(1,9),(5,10)
0,3,(2,7),(5,7)
0,3,(3,5),(6,9)
0,3,(3,6),(5,8)
0,3,(3,7),(7,10)
0,4,(1,2),(9,10)
0,4,(2,3),(2,10)
0,4,(3,7),(4,8)
0,5,(1,4),(6,9)
0,5,(2,8),(6,8)
0,5,(4,7),(6,7)
0,1,2,3,5,(4,6)
0,1,2,4,5,(4,6)
0,1,2,4,7,(2,3)
0,1,2,4,7,(4,9)
0,1,2,4,7,(8,9)
0,1,2,4,10,(1,9)
0,1,2,4,10,(3,9)
0,1,2,7,8,(1,9)
0,1,2,7,8,(9,10)
0,1,3,4,10,(6,10)
0,1,3,6,8,(6,8)
0,1,3,6,10,(7,9)
0,1,3,7,9,(1,8)
0,1,4,5,8,(5,7)
0,1,4,7,10,(1,9)
0,1,5,6,8,(5,9)
0,1,5,7,9,(2,8)
0,1,6,8,9,(2,6)
0,2,3,7,8,(4,10)
0,2,3,7,8,(6,10)
0,2,3,7,8,(7,10)
0,2,4,5,9,(5,9)
0,3,4,5,6,(2,10)
0,3,4,6,7,(2,3)
0,3,5,6,7,(4,8)

n = 12:
0,3,8,(3,9)
0,4,7,(1,7)
0,4,7,(4,7)
0,1,(2,3),(3,4)
0,1,(2,5),(3,10)
0,1,(2,8),(6,10)
0,1,(7,8),(8,10)
0,1,(8,11),(9,10)
0,2,(1,3),(3,6)
0,2,(1,7),(2,8)
0,2,(1,10),(1,11)
0,2,(2,3),(7,9)
0,2,(3,9),(3,11)
0,2,(3,9),(5,9)
0,2,(5,11),(8,11)
0,2,(7,9),(7,11)
0,3,(1,8),(7,10)
0,3,(5,11),(6,10)
0,1,2,3,5,(5,9)
0,1,2,5,9,(7,11)
0,1,2,6,11,(2,6)
0,1,3,6,7,(4,10)
0,1,3,6,9,(1,9)
0,1,3,6,9,(4,10)
0,1,3,7,10,(4,5)
0,1,4,8,10,(2,5)
0,1,5,6,8,(4,6)
0,1,5,6,8,(6,10)
0,1,5,6,11,(7,8)
0,1,5,7,9,(1,11)
0,1,5,9,10,(6,7)
0,2,3,4,10,(3,8)
0,2,3,6,8,(3,6)
0,2,3,6,10,(2,6)
0,2,3,6,10,(4,10)
0,2,5,6,10,(2,10)

n = 13:
0,1,11,(5,9)
0,4,8,(9,10)
0,1,(1,7),(3,7)
0,1,(2,3),(6,11)
0,1,(2,5),(5,11)
0,1,(2,6),(6,8)
0,1,(2,9),(4,5)
0,2,(1,6),(9,12)
0,2,(7,10),(10,12)
0,3,(1,9),(2,11)
0,3,(4,6),(9,11)
0,3,(8,9),(9,10)
0,4,(1,3),(4,6)
0,4,(1,3),(10,12)
0,4,(2,9),(8,10)
0,5,(1,5),(4,9)
0,5,(1,12),(7,11)
0,5,(2,9),(4,5)
0,5,(3,6),(4,9)
0,5,(3,12),(9,11)
0,6,(1,5),(2,12)
0,1,2,4,5,(1,7)
0,1,2,10,11,(6,12)
0,1,3,4,6,(6,10)
0,1,4,5,10,(4,8)
0,1,5,6,7,(5,9)
0,1,5,7,9,(8,9)
0,1,5,7,11,(8,10)
0,1,7,10,11,(2,6)
0,1,8,9,10,(8,9)
0,2,3,8,11,(1,10)
0,2,5,6,11,(8,11)
0,2,6,7,10,(8,12)
0,3,4,5,12,(4,5)
0,3,5,6,10,(8,11)
0,3,5,7,10,(2,10)

n = 14:
0,1,2,(7,12)
0,1,(2,13),(4,12)
0,1,(5,12),(9,12)
0,2,(1,5),(3,11)
0,3,(1,6),(4,12)
0,3,(2,4),(6,12)
0,3,(2,12),(6,13)
0,3,(5,10),(7,12)
0,5,(2,4),(6,13)
0,6,(1,13),(5,9)
0,6,(5,9),(12,13)
0,1,2,3,5,(1,3)
0,1,2,4,7,(1,3)
0,1,4,5,8,(2,8)
0,1,4,5,13,(1,6)
0,1,4,7,11,(1,11)
0,1,6,10,12,(3,7)
0,1,6,10,12,(7,9)
0,1,7,9,12,(3,13)
0,2,3,5,7,(1,5)
0,2,3,10,12,(9,10)
0,2,5,6,12,(6,10)
0,2,7,9,11,(11,12)
0,4,5,6,8,(1,4)
0,4,6,7,10,(5,13)

n = 15:
0,5,9,(2,11)
0,2,(6,8),(12,14)
0,4,(2,11),(7,10)
0,4,(5,6),(5,14)
0,4,(6,10),(9,10)
0,4,(7,8),(12,14)
0,6,(8,11),(12,13)
0,7,(2,11),(10,13)
0,7,(3,12),(3,13)
0,1,3,7,11,(9,10)
0,1,4,5,12,(3,4)
0,1,4,6,11,(2,14)
0,1,4,9,10,(7,10)
0,1,5,11,13,(5,11)
0,2,3,9,10,(6,10)
0,2,3,9,13,(3,7)
0,2,4,10,14,(4,10)
0,3,4,5,10,(3,7)
0,3,5,7,8,(3,13)
0,4,5,7,10,(1,14)
0,4,8,12,14,(5,6)
0,4,9,11,14,(1,13)
0,5,6,11,14,(5,8)
0,5,6,12,13,(5,9)

n = 16:
0,2,13,(2,3)
0,3,(1,5),(5,7)
0,3,(2,13),(7,14)
0,5,(4,8),(6,12)
0,5,(4,12),(7,8)
0,7,(2,6),(10,13)
0,7,(8,14),(11,12)
0,1,2,3,9,(6,14)
0,1,5,13,14,(14,15)
0,1,11,12,13,(5,15)
0,2,5,10,14,(6,14)
0,2,6,11,12,(14,15)
0,2,7,8,10,(3,6)
0,2,7,8,13,(3,15)
0,4,8,9,10,(8,12)

n = 17:
0,1,(7,10),(9,15)
0,3,(6,9),(13,14)
0,5,(4,7),(6,13)
0,6,(2,9),(7,12)
0,7,(1,8),(9,14)
0,8,(10,12),(11,16)
0,1,3,9,12,(7,13)
0,1,3,12,14,(2,10)
0,1,5,9,11,(1,13)
0,1,7,11,13,(6,14)
0,2,4,9,12,(6,16)
0,3,6,7,10,(9,15)
0,3,8,11,12,(3,11)
0,4,6,10,16,(3,11)
0,5,6,9,14,(6,14)

n = 18:
0,1,(9,12),(11,12)
0,3,(2,11),(6,7)
0,1,4,5,7,(2,10)
0,1,5,11,12,(1,9)
0,1,8,9,11,(2,15)
0,2,6,12,15,(11,15)
0,3,9,11,13,(4,16)

n = 19:
0,7,10,(6,18)
0,9,12,(1,13)
0,2,(6,8),(8,10)
0,4,(5,16),(7,14)
0,6,(4,8),(17,18)
0,1,4,5,8,(5,15)
0,1,4,8,17,(1,13)
0,3,7,9,16,(3,17)
0,5,6,12,14,(2,18)

n = 20:
0,4,6,7,15,(3,7)
0,6,7,12,14,(4,6)

n = 21:
0,1,15,17,19,(13,15)
0,2,7,12,17,(4,10)
0,3,5,9,13,(15,17)
0,4,8,9,11,(3,11)

n = 22:
0,1,(4,10),(11,18)
0,5,(4,12),(7,14)
0,1,6,8,12,(10,17)
0,1,10,16,18,(3,21)
0,5,6,11,15,(9,21)

n = 23:
0,3,(13,19),(18,19)
0,2,6,10,14,(5,13)
0,3,11,16,18,(4,19)

n = 24:
0,1,8,9,15,(7,18)

5 Conclusion

We presented a complete list of n-bit NLFSRs with the period $2^n - 1$, $n < 25$, for three different types of feedback functions with the algebraic degree two. This list is available at http://web.it.kth.se/∼dubrova/nlfsr.html.

6 Acknowledgement

This work was supported in part by a research grant 621-2010-4388 from the Swedish Research Council.

References

1. F. S. Annexstein. Generating de Bruijn sequences: An efficient implementation. *IEEE Transactions on Computers*, 46:198 – 200, 1997.
2. J.-M. Chabloz, S. Mansouri, and E. Dubrova. An algorithm for constructing a fastest Galois NLFSR generating a given sequence. In C. Carlet and A. Pott, editors, *Sequences and Their Applications - SETA 2010*, volume 6338 of *Lecture Notes in Computer Science*, pages 41–54. Springer Berlin / Heidelberg, 2010.
3. T. Chang, B. Park, Y. H. Kim, and I. Song. An efficient implementation of the D-homomorphism for generation of de bruijn sequences. *IEEE Transactions on Information Theory*, 45:1280–1283, 1999.
4. T. Chang, I. Song, and S. H. Cho. Some properties of cross-join pairs in maximum length linear sequences. In *Proceeding of ISZTA '90*, pages 1077–1079, 1990.
5. T. W. Cusick and P. Stănică. *Cryptographic Boolean functions and applications*. Academic Press, San Diego, CA, USA, 2009.
6. N. G. de Bruijn. A combinatorial problem. *Nederl. Akad. Wetensch*, 49:758–746, 1946.
7. R. Diestel. *Graph Theory*. Springer-Verlag, 1997.
8. E. Dubrova. A transformation from the Fibonacci to the Galois NLFSRs. *IEEE Transactions on Information Theory*, 55(11):5263–5271, November 2009.
9. E. Dubrova. Finding matching initial states for equivalent NLFSRs in the Fibonacci to the Galois configurations. *IEEE Transactions on Information Theory*, 56(6):2961–2967, June 2010.
10. E. Dubrova. A scalable method for constructing Galois NLFSRs with period $2^n - 1$ using cross-join pairs. Technical Report 2011/632, Cryptology ePrint Archive, November 2011. http://eprint.iacr.org/2011/632.
11. E. Dubrova, M. Teslenko, and H. Tenhunen. On analysis and synthesis of (n,k)-non-linear feedback shift registers. In *Design and Test in Europe*, pages 133–137, 2008.
12. T. Etzion and A. Lempel. Algorithms for the generation of full-length shift register sequences. *IEEE Transactions on Information Theory*, 3:480–484, May 1984.
13. H. Fredricksen. A class of nonlinear deBruijn cycles. *J. Comb. Theory*, 19(A):192–199, Sept. 1975.
14. H. Fredricksen. A survey of full length nonlinear shift register cycle algorithms. *SIAM Review*, 24(2):195–221, 1982.
15. H. M. Fredricksen. Disjoint cycles from de Bruijn graph. Technical Report 225, USCEE, 1968.
16. H. M. Fredricksen and I. J. Kessler. Lexicographic compositions and de Bruijn sequences. *J. Comb. Theory*, 22:17–30, 1977.

17. H. M. Fredricksen and J. Maiorana. Neklaces of beads in k colors and k-ary de Bruijn sequences. *Discrete Math.*, 23:207–210, 1978.
18. S. Golomb. *Shift Register Sequences*. Aegean Park Press, 1982.
19. T. Helleseth and T. Kløve. The number of cross-join pairs in maximum length linear sequences. *IEEE Transactions on Information Theory*, 31:1731–1733, 1991.
20. I. Janicka-Lipska and J. Stoklosa. Boolean feedback functions for full-length nonlinear shift registers. *Telecommunications and Informatioin Technology*, 5:28–29, 2004.
21. C. J. Jansen. *Investigations On Nonlinear Streamcipher Systems: Construction and Evaluation Methods*. Ph.D. Thesis, Technical University of Delft, 1989.
22. C. J. A. Jansen and D. E. Boekee. The shortest feedback shift register that can generate a given sequence. In *Proceedings on Advances in cryptology*, CRYPTO '89, pages 90–99, New York, NY, USA, 1989. Springer-Verlag New York, Inc.
23. E. J. v. Lantschoot. Double adjacencies between cycles of a circulating shift register. *Transactions on Computers*, C-22:944–954, 1973.
24. R. Lidl and H. Niederreiter. *Introduction to Finite Fields and their Applications*. Cambridge Univ. Press, 1994.
25. K. Limniotis, N. Kolokotronis, and N. Kalouptsidis. On the nonlinear complexity and Lempel-Ziv complexity of finite length sequences. *IEEE Transactions on Information Theory*, 53(11):4293–4302, 2007.
26. D. Linardatos and N. Kalouptsidis. Synthesis of minimal cost nonlinear feedback shift registers. *Signal Process.*, 82(2):157–176, 2002.
27. J. L. Massey. Shift-register synthesis and BCH decoding. *IEEE Transactions on Information Theory*, 15:122–127, 1969.
28. E. McCluskey. Built-in self-test techniques. *IEEE Design and Test of Computers*, 2:21–28, 1985.
29. J. McCluskey. High speed calculation of cyclic redundancy codes. In *Proceedings of the 1999 ACM/SIGDA seventh international symposium on Field programmable gate arrays*, FPGA '99, pages 250–256, New York, NY, USA, 1999. ACM.
30. G. Mrugalski, J. Rajski, and J. Tyszer. Ring generators - New devices for embedded test applications. *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 23(9):1306–1320, 2004.
31. J. Mykkelveit. Generating and counting the double adjacencies in a pure cycling shift register. *Transactions on Computers*, C-24:299–304, 1975.
32. A. Ralston. A new memoryless algorithm for de Bruijn sequences. *J. Algorithms*, 2:50–62, 1981.
33. E. Roth. Permitations arranged around a cycle. *Amer. Math. Monthly*, pages 990–992, 1971.
34. K. Zeng, C. Yang, D. Wei, and T. R. N. Rao. Pseudo-random bit generators in stream-cipher cryptography. *Computer*, 1991.