Cryptographic D-morphic Analysis and Fast Implementations of Composited De Bruijn Sequences

Kalikinkar Mandal, and Guang Gong

Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, N2L 3G1, CANADA
{`kmandal, ggong`}`@uwaterloo.ca`

Abstract. Recently, Mandal and Gong [24] refined and analyzed the composited construction for generating de Bruijn sequences, where the composited feedback function is the sum of a feedback function with k-th order composition and a sum of $(k+1)$ product-of-sum terms. In this paper we first determine the linear complexity of a composited de Bruijn sequence. We then conduct a profound analysis of the recursive construction by introducing the notion of the higher order D-morphism of a binary sequence. In the analysis, we consider both linearly and nonlinearly generated composited de Bruijn sequences, and show the existence of k-th order D-morphic order n de Bruijn preimages ((n,k)-DMDPs) of length (2^n+k) and k-th order D-morphic order n m-sequence preimages ((n,k)-DMMPs) of length $(2n+k)$ in a nonlinearly and linearly generated composited de Bruijn sequence, respectively. We show that an (n,k)-DMMP/DMDP is a vulnerable segment of a composited de Bruijn sequence as an (n,k)-DMMP/DMDP allows one to reconstruct the composited de Bruijn sequence. We determine the exact number of (n,k)-DMDPs in a composited de Bruijn sequence, and calculate the success probability of obtaining an (n,k)-DMMP/DMDP. Furthermore, we present a new iterative technique with its parallel extension for computing the product-of-sum terms of the feedback function where a product-of-sum term is calculated in an iteration. In addition, we present three de Bruijn sequences of period 2^{64} together with their software implementations and performances.

Keywords: Nonlinear feedback shift registers, De Bruijn sequences, Span n sequences, Pseudorandom sequence generators, Software implementations.

1 Introduction

Due to limited theoretical results in the theory of nonlinear feedback shift registers, current methodologies of using a nonlinear feedback shift register (NLFSR) in a stream cipher or a pseudorandom sequence generator are to mask with a linear feedback shift register (LFSR) or another NLFSR. In such a design, if the LFSR works independently, without any feedback from NLFSRs, then the design can promise the lower bound of the periodicity of a keystream. Otherwise, the lower bound of the periodicity of a keystream cannot be promised. A number of NLFSR-based stream cipher proposals can be found in the ECRYPT project [8]. For example, stream cipher Grain which can guarantee the lower bound of the period whereas stream ciphers Trivium and Mickey cannot guarantee the lower bound of the period for any initial state [8].

A binary *de Bruijn sequence* is a sequence of period 2^n where each n-tuple occurs exactly once in one period of the sequence (this is referred to as the *span n property*). A de Bruijn sequence generated by an n-stage NLFSR has known randomness properties such as long period, balance, span n property [7, 12, 13]. The *linear complexity or linear span* of a sequence is defined as the length of the shortest LFSR which generates the sequence. Moreover, de Bruijn sequences have high linear complexity, i.e., the linear complexity is lower bounded by half of its period [4]. A *modified de Bruijn* or *span n sequence* can be obtained by deleting one zero bit from the run of zeros of length n of a de Bruijn sequence of period 2^n. Although a span n sequence preserves the balance property and span n properly of the corresponding de Bruijn sequence, the linear span of the span n sequence can be very low. Any (low linear complexity) span n sequence (e.g. m-sequence) can be made to a de Bruijn sequence by inserting one zero to the run of zeros of length $(n-1)$ where the linear complexity of this de Bruijn sequence is bounded below by $(2^{n-1}+n)$ [4]. Again, from this de Bruijn sequence, one can remove a zero from the run of zeros of length n then it becomes a span n sequence of the original linear complexity. Thus, the lower bound of the linear complexity of this de Bruijn sequence varies drastically only after removing one zero from the run of zeros of length n [16]. This shows that the linear complexity of a de Bruijn sequence is not an adequate measurement for its randomness. Instead, it should be measured in terms of the linear complexity of its corresponding span n sequence, since they have only one bit difference [24].

Several publications in the literature have been discussed various techniques of producing de Bruijn sequences [1, 3, 9–11, 20, 22–24, 27]. There are two main challenges for the production of de Bruijn sequences. One challenge is to produce plenty of de Bruijn sequences of the same length for different recurrence relations whereas another one aims to produce long de Bruijn sequences which can have many practical applications in cryptography to design stream ciphers and PRSGs. Most of the existing techniques are concerned about the production of different de Bruijn sequences of the same period and which are not efficient for generating long period de Bruijn sequences [1, 3, 9, 10, 20]. Only a few recent publications consider the production of long period de Bruijn sequences [23, 24, 28].

Very recently, Mandal and Gong [24] refined and analyzed the method of generating a large period strong de Bruijn sequence by Lempel [22] and Mykkelvit *et al.* [27] where an $(n+k)$-stage NLFSR that generates a de Bruijn sequence is constructed through composition operation from an n-stage NLFSR that generates a span n sequence. As an analysis, they showed an approximation of the feedback function by the k-th order composition of the feedback function for the span n sequence where the product-of-sum terms are set to the zero function due to their low hamming weights. They also determined the cycle structure of the approximated recurrence relation.

In the composition method, a de Bruijn sequence of order $(n+k)$ is constructed by first finding two D-morphic preimages of a de Bruijn sequence of order $(n+k-1)$ and then concatenating these two preimages at the conjugate pair. Our contributions in this paper are:

1. We first determine the linear complexity of a composited de Bruijn sequence which is much greater than the lower bound $(2^{n+k-1}+n+k)$ of the linear complexity of any de Bruijn sequence of period 2^{n+k}.
2. We then make a profound analysis of the composited de Bruijn sequences by introducing the notion of higher order D-morphisms and higher order D-morphic preimages. We consider both linearly and nonlinearly generated de Bruijn sequences. Because of the recursive construction, there exist a number of k-th order D-morphic order n de Bruijn preimages ((n,k)-DMDPs in brief) of length (2^n+k) and k-th order D-morphic order n m-sequence preimages ((n,k)-DMMPs in brief) of length $(2n+k)$ in a nonlinearly and linearly generated de Bruijn sequence of period 2^{n+k}, respectively. We show that an (n,k)-DMDP/DMMP is a vulnerable segment and it allows one to recover a composited de Bruijn sequence with low complexity. We determine the exact number of (n,k)-DMDPs in a composited de Bruijn sequence and calculate the success probability of obtaining an (n,k)-DMDP/DMMP for both cases.

3. We propose a new iterative technique for computing the product-of-sum terms of the feedback function, which complies to evaluate all product-of-sum terms in k iterations. To reduce the number of iterations, we give a parallel extension of the iterative technique for evaluating all the product-of-sum terms.
4. Finally, we present three instances of 64-stage NLFSRs for different parameters and show their performances in software implementation.

The paper is organized as follows. In Section 2, we define and explain some notations and terms and briefly review the recursive construction of an NLFSR that can generate a de Bruijn sequence. Section 3 determines the linear complexity of composited de Bruijn sequences. In Section 4, we perform a deep analysis of the recursive construction using higher order D-morphisms. In Section 5, a new iterative method and its parallel extension for computing the product-of-sum terms of the feedback function are proposed. In Section 6, we present three 64-stage NLFSRs that generate de Bruijn sequences with period 2^{64} together with their performances in software implementations. Finally, in Section 7, we conclude the paper.

2 Background

In this section, we define and explain the notations, terms and mathematical functions that will be used in this paper. Moreover, we review the recursive construction for generating long period de Bruijn sequences.

- $\mathbb{F}_2 = \{0, 1\}$: the Galois field with two elements.
- $\mathbb{F}_{2^t}$: a finite field with 2^t elements that is defined by a primitive element α with $p(\alpha) = 0$, where $p(x) = c_0 + c_1x + \cdots + c_{t-1}x^{t-1} + x^t$ is a primitive polynomial of degree t (≥ 2) over $\mathbb{F}_2$.
- Z_o^n and Z_e^n denote two sets of odd integers and even integers between 1 and n, respectively.
- $\psi(x_0, x_1) = x_0 + x_1$: a Boolean function.
- $e_i = (1, 1, 0, 1, 0, ...) \in \mathbb{F}_{2^i}$ and $\hat{e}_i = (0, 1, 0, 1, 0, ...) \in \mathbb{F}_{2^i}$ are the conjugate pair for an i-stage NLFSR.

2.1 Definition of FSRs and Their Compositions

Usually, a periodic binary sequence $\mathbf{a} = \{a_i\}$ is generated by an n-stage (non)linear feedback shift register, which is defined as [13]

$$a_{n+k} = f(a_k, a_{k+1}, ..., a_{k+n-1}) = a_k + G(a_{k+1}, ..., a_{k+n-1}),\ a_k \in \mathbb{F}_2,\ k \geq 0$$

where $(a_0, a_1, ..., a_{n-1})$ is called the *initial state* of the feedback shift register, f is a Boolean function in n variables, and G is a Boolean function in $(n-1)$ variables. If the function f is an affine function, then the sequence $\mathbf{a}$ is called an *LFSR sequence*; otherwise it is called an *NLFSR sequence.*

Let $g(x_0, x_1, ..., x_n) = x_0 + x_n + G(x_1, x_2, ..., x_{n-1})$ be a Boolean function in $(n + 1)$ variables where G is a Boolean function in $(n - 1)$ variables. The *first order* composition of ψ and g, denoted as $g \circ \psi$, is defined by [27]

$$g \circ \psi = g(x_0 + x_1, x_1 + x_2, ..., x_n + x_{n+1}) = x_0 + x_1 + x_{n+1} + x_n + G(x_1 + x_2, ..., x_{n-1} + x_n).$$

Similarly, the *k-th order* composition of g (with respect to ψ) is defined by $g \circ \psi^k = \left(g \circ \psi^{k-1}\right) \circ \psi$, where $g \circ \psi^{k-1}$ is the $(k-1)$-th order composition of g with respect to ψ [24].

2.2 The D-morphisms of Binary Sequences

Let $\mathbf{a} = (a_0, a_1, a_2, ..., a_N)$ be a binary sequence of length $N(\geq 1)$. In [22], Lempel introduced the idea of the D homomorphism (D-morphism) and its inverse of a binary sequence. The *first order* D-morphic image of $\mathbf{a}$ is defined as [22]

$$D(\mathbf{a}) = (a_0 + a_1, a_1 + a_2, a_2 + a_3, ..., a_{N-1} + a_N).$$

Again, the D-morphic preimages of $\mathbf{a}$ are given by [22]

$$z = (z_i) = (0, a_0, a_0 + a_1, a_0 + a_1 + a_2, ..., \sum_{i=0}^{2^n-1} a_i) \text{ and } \bar{z} = (\bar{z}_i), \bar{z}_i = z_i + 1.$$

2.3 Description of the Recursive Construction

In this subsection we first recollect the refined construction for an arbitrary stage recursive nonlinear feedback function from [24]. We use the notations of [24] for describing the construction briefly.

Denote $X_0^p = \prod_{i \in Z_o^p} x_i \prod_{i \in Z_e^p} (x_i + 1)$. The k-th order composition of X_0^p, denoted by X_k^p, is defined as $X_k^p = \prod_{i \in Z_o^p} (x_i \circ \psi^k) \prod_{i \in Z_e^p} (x_i \circ \psi^k + 1)$. Note that X_k^p is a *product-of-sum term*, which is a Boolean function in $(p + k)$ variables with Hamming weight 2^k and the expansion of $(x_i \circ \psi^k)$ contains 2^l variables, where l is the Hamming weight of k.

A function I_k^n of algebraic degree $(n + k - 1)$ is defined by summing up all $X_i^p, (i + p) = (n + k - 1),\ 0 \leq i \leq k - 1$ and J_k^{n-1} as

$$I_k^n(x_1, x_2, ..., x_{n+k-1}) = J_k^{n-1} + X_{k-1}^n + X_{k-2}^{n+1} + \cdots + X_0^{n+k-1}$$

where $J^{n-1} = \prod_{i=1}^{n-1}(x_i + 1)$ and $J_k^{n-1} = \prod_{i=1}^{n-1}(x_i \circ \psi^k + 1)$ [24]. The function I_k^n is a function of $(k + 1)$ product-of-sum terms. The construction of an $(n + k)$-stage NLFSR that generates a de Bruijn sequence is stated in the following lemma.

Lemma 1. *[24] Let $g(x_0, x_1, ..., x_n) = x_n + x_0 + G(x_1, x_2, ..., x_{n-1})$, which generates a span n sequence of period $(2^n - 1)$, where G is a Boolean function in $(n - 1)$ variables. Then, for any integer $k \geq 0$,*

$$R_k^n(x_0, x_1, ..., x_{n+k}) = (x_n + x_0) \circ \psi^k + G(x_1, x_2, ..., x_{n-1}) \circ \psi^k + I_k^n(x_1, ..., x_{n+k-1}) = 0 \quad (1)$$

generates a de Bruijn sequence of period 2^{n+k}.

We define $h = g + J^{n-1}$, which is a function in n variables. In the above lemma, h is the feedback function which generates the *starting de Bruijn sequence* of order n. A de Bruijn sequence generated by recurrence relation (1) is called a *composited de Bruijn sequence*, and the feedback function $R_k^n = 0$ is called a composited feedback function.

The Welch-Gong (WG) Transformation. Let t be a positive integer with $t \not\equiv 0 \bmod 3$ and $3k \equiv 1 \bmod t$ for some integer k. Let $\mathrm{Tr}(x) = x + x^2 + \cdots + x^{2^{t-1}}, x \in \mathbb{F}_{2^t}$ be the trace function mapping from $\mathbb{F}_{2^t}$ to $\mathbb{F}_2$. Then the function, from $\mathbb{F}_{2^t}$ to $\mathbb{F}_2$, defined by

$$f(x^d) = \mathrm{Tr}(h(x^d + 1) + 1)$$

is known as the *WG transformation* with d decimation [14, 15], where d is a coset leader which is co-prime with $2^t - 1$, $h(x) = x + x^{q_1} + x^{q_2} + x^{q_3} + x^{q_4}$ and $q_1 = 2^k + 1, q_2 = 2^{2k} + 2^k + 1, q_3 = 2^{2k} - 2^k + 1, q_4 = 2^{2k} + 2^k - 1$. The WG transformation has good cryptographic properties such as high algebraic degree, high nonlinearity and a WG sequence has high linear span.

3 Linear Complexity of Composited de Bruijn Sequences

This section determines the linear complexity of a composited de Bruijn sequence produced by a composited nonlinear recurrence relation in which the linear complexity of the starting span n sequence is known. In this paper, we use the de Bruijn sequence of order n and the de Bruijn sequence of period 2^n interchangeably.

3.1 A Closer Look at the Composited Construction

Let $\mathbf{s}$ be a de Bruijn sequence of order $(n + 1)$ produced by the recurrence relation (1) when $k = 1$. The composited construction of a de Bruijn sequence of order $(n+k)$ or the recurrence relation (1) can be interpreted as follows. Let $\mathbf{a} = \{a_i\}_{i \geq 0}$ be a de Bruijn sequence of order n that is generated by $h = g + J^{n-1}$. Then the D-morphic preimages of $\mathbf{a}$ are z and $\bar{z}$, given in Section 2.2. According to the recurrence relation (1) for $k = 1$, the sequence $\mathbf{s}$ can be written as $\mathbf{s} = z \| E^t(\bar{z})$, where $z = z' \| e$, $\bar{z} = z'' \| \hat{e}$, $0 \leq t \leq 2^n - 1$, E is the left shift operator, and $\|$ denotes the concatenation operation [4].

We denote by $\mathbf{s}_i = z_i \| E^{t_i}(\bar{z}_i)$ the de Bruijn sequence of order $(n+i)$ for $0 \leq t_i \leq 2^{n+i-1} - 1$ and $\mathbf{s}_0 = \mathbf{a}$. A de Bruijn sequence of order $(n + k)$ is constructed recursively by calculating preimages as follows:

$$\begin{aligned} \mathbf{s}_1 &= z_1 \| E^{t_1}(\bar{z}_1) \text{ for } 0 \leq t_1 \leq 2^n - 1 \\ \mathbf{s}_2 &= z_2 \| E^{t_2}(\bar{z}_2) \text{ for } 0 \leq t_2 \leq 2^{n+1} - 1 \\ \vdots \quad & \quad \vdots \\ \mathbf{s}_k &= z_k \| E^{t_k}(\bar{z}_k) \text{ for } 0 \leq t_k \leq 2^{n+k-1} - 1 \end{aligned}$$

where z_i and $\bar{z}_i$ are D-morphic preimages of the de Bruijn sequence $\mathbf{s}_{i-1}$. This is an equivalence between the recurrence relation (1) and the construction of a de Bruijn sequence of order $(n + k)$ from a de Bruijn sequence of order n when the concatenation is performed at the conjugate pair $e_i = (1, 1, 0, 1, 0, ..., 1, 0) \in \mathbb{F}_{2^i}$ and $\hat{e}_i = (0, 1, 0, 1, 0, ..., 1, 0) \in \mathbb{F}_{2^i}$, $n \leq i \leq n + k - 1$.

Remark 1. Annexstein [1] proposed an algorithm for generating a long de Bruijn sequence by computing preimages of lower order de Bruijn sequences. The algorithm is similar to the above representation. Chang *et al.* [6] proposed another algorithm based on D-homomorphism for producing a long de Bruijn sequence from a short de Bruijn sequence.

3.2 Linear Complexity of a Composited de Bruijn Sequence

We now determine the linear complexity of a de Bruijn sequence produced by recurrence relation (1) in terms of the linear complexity of the starting span n sequence generated by g. We use the notations used in Section 3.1 in the following theorem.

Theorem 1. *Let the linear complexity of a span n sequence generated by g be optimal, i.e, $2^n - 2$. Then the linear complexity of a de Bruijn sequence $\boldsymbol{s}_k$ of period 2^{n+k} generated by recurrence relation (1), denoted as $LC(\boldsymbol{s}_k)$, is bounded below by $(2^{n+k} - 2 - \sum_{i=1}^{k} 2^{m_i})$ where $2^{m_i} \mid t_i$ but $2^{m_i+1} \nmid t_i$, $1 \leq i \leq k$.*

Proof. For $k = 1$, the de Bruijn sequence $\mathbf{s}_1$ can be written as $\mathbf{s}_1 = z_1 \| E^{t_1}(\bar{z}_1)$ for $0 \leq t_1 \leq 2^n - 1$, where $z_1 = z_1' \| e$, $\bar{z}_1 = z_1'' \| \hat{e}$. By Theorem 11 of [4],

$$LC(\mathbf{s}_1) \geq 2^n + 2^n - 2 - 2^{m_1} = 2^{n+1} - 2 - 2^{m_1}$$

where $2^{m_1} \mid t_1$ but $2^{m_1+1} \nmid t_1$ as $LC(\mathbf{s}_0)$ is greater than or equal to the linear complexity of the starting span n sequence generated by g. As de Bruijn sequence $\mathbf{s}_2$ is constructed from $\mathbf{s}_1$ in the same way, applying the same argument, the linear complexity of sequence $\mathbf{s}_2$ is

$$LC(\mathbf{s}_2) \geq 2^{n+1} + 2^{n+1} - 2 - 2^{m_1} - 2^{m_2} = 2^{n+2} - 2 - 2^{m_1} - 2^{m_2}$$

where $2^{m_2} \mid t_2$ but $2^{m_2+1} \nmid t_2$. In general, for $k \geq 1$, the linear complexity bound of $\mathbf{s}_k$ is

$$LC(\mathbf{s}_k) \geq 2^{n+k} - 2 - \sum_{i=1}^{k} 2^{m_i}$$

where $2^{m_i} \mid t_i$ but $2^{m_i+1} \nmid t_i$, $1 \leq i \leq k$. □

Since the exact linear complexity of a composited de Bruijn sequence depends on the values of m_i's, we computed the linear complexity of many composited de Bruijn sequences when the starting span n sequences generated by g have optimal or near-optimal linear complexity for $(n+k) = 11, 12, ...,$ and 20 and for different values of k and n. Our experimental result shows that the linear complexities of composited de Bruijn sequences are optimal or close to optimal, both of which are much greater than the lower bound $(2^{n+k-1} + n + k)$.

Remark 2. If L $(\geq 2^{n-1} + 2)$ is the linear complexity of a span n sequence generated by g, then the linear complexity of $\mathbf{s}_k$ satisfies $LC(\mathbf{s}_k) \geq L + 2^n(2^k - 1) - \sum_{i=1}^{k} 2^{m_i}$.

4 Vulnerability of Higher Order *D*-Morphic Preimages in Composited de Bruijn Sequences

As a composited de Bruijn sequence of period 2^{n+k} is obtained from a de Bruijn sequence of period 2^n through the composition operation by the recurrence relation (1), the problem of investigating the reconstruction of a composited de Bruijn sequence and the recovery of its feedback function is crucial for analyzing the security of the composited construction. In this section, we consider both linearly and nonlinearly generated de Bruijn sequences and show the existence of many k-th order D-morphic order n de Bruijn preimages ((n, k)-DMDPs) and k-th order D-morphic order n m-sequence preimages ((n, k)-DMMPs). We also calculate the number of (n, k)-DMDPs in a composited de Bruijn sequence and the success probability of obtaining an (n, k)-DMDP from a composited de Bruijn sequence of order $(n + k)$. Moreover, we show that an (n, k)-DMDP is a vulnerable segment, and it allows one to reconstruct the de Bruijn sequence.

4.1 Higher Order D-morphisms and Enumeration of (n,k)-DMDPs

In this section, we first define higher order D-morphic preimages of a binary sequence. Then we calculate the number of (n,k)-DMDPs in a composited de Bruijn sequence of period 2^{n+k}.

Higher Order D-morphic Preimages. Let $\mathbf{a} = (a_0, a_1, a_2, ..., a_{N-1})$ be a binary sequence of length $N (\geq 1)$. The first order D-morphic image of $\mathbf{a}$ in terms of the composition operation ψ can be written as

$$D(\mathbf{a}) = (a_0 \circ \psi, a_1 \circ \psi, a_2 \circ \psi, ..., a_{N-2} \circ \psi).$$

Similarly, the *k-th order* D-morphic image of $\mathbf{a}$ in terms of ψ, denoted by $D^k(\mathbf{a})$, is defined as

$$D^k(\mathbf{a}) = (a_0 \circ \psi^k, a_1 \circ \psi^k, a_2 \circ \psi^k, ..., a_{N-k-1} \circ \psi^k).$$

Example 1. Let $\boldsymbol{b} = (b_0, b_1, b_2, b_3, b_4, b_5, b_6)$ be a binary sequence of length 7. The 3-order D-morphic image of $\boldsymbol{b}$ is given by

$$D^3(\boldsymbol{b}) = (b_0 + b_1 + b_2 + b_3, b_1 + b_2 + b_3 + b_4, b_2 + b_3 + b_4 + b_5, b_3 + b_4 + b_5 + b_6).$$

We now generalize the notion of higher order D-morhic preimages of a binary sequence.

Definition 1. *A segment $\boldsymbol{s}$ of length $(N+k)$ is called* a k-th order D-morphic preimage *of $\boldsymbol{a}$ if $D^k(\boldsymbol{s}) = \boldsymbol{a}$.*

Proposition 1. *For $k \geq 1$, the total number of distinct k-th order D-morphic preimages of a binary sequence is equal to 2^k.*

Example 2. The second order D-morphic preimages of $\mathbf{a} = \{a_0, a_1, a_2, a_3, a_4, a_5\}$ are given by

$$\begin{aligned}
z_0 &= (0, 0, a_0, a_1, a_0 + a_2, a_1 + a_3, a_0 + a_2 + a_4, a_1 + a_3 + a_5)\\
z_1 &= (0, 1, a_0, 1 + a_1, a_0 + a_2, 1 + a_1 + a_3, a_0 + a_2 + a_4, 1 + a_1 + a_3 + a_5)\\
z_2 &= (1, 0, 1 + a_0, a_1, 1 + a_0 + a_2, a_1 + a_3, 1 + a_0 + a_2 + a_4, a_1 + a_3 + a_5)\\
z_3 &= (1, 1, 1 + a_0, 1 + a_1, 1 + a_0 + a_2, 1 + a_1 + a_3, 1 + a_0 + a_2 + a_4, 1 + a_1 + a_3 + a_5).
\end{aligned}$$

We see that z_1, z_2 and z_3 can be obtained from z_0 by preforming bitwise addition between z_0 and $\{0,1,0,1...\}$, $\{1,0,1,0...\}$ and $\{1,1,1,1...\}$, respectively. For $k \geq 2$, it is sufficient to compute only z_0 iteratively, and other $2^k - 1$ preimages can be obtained easily from z_0.

When a k-th D-morphic image of a binary sequence of length N is calculated, the length of the k-th D-morphic image becomes $(N-k)$ as at each iteration the length is reduced by one. On the other hand, the length of a D-morphic preimage of a sequence is increased by one when one-order D-morphic preimage of the sequence is computed, and the length of a k-th order D-morphic preimage of a binary sequence of length N is $(N+k)$.

Definition 2. *A segment $\boldsymbol{s}$ of length $(2^n + k)$ of a composited de Bruijn sequence of period 2^{n+k} is called a k*-th order D-morphic order n de Bruijn preimage *((n,k)-DMDP) if $D^k(\boldsymbol{s})$ is the starting de Bruijn sequence of period 2^n.*

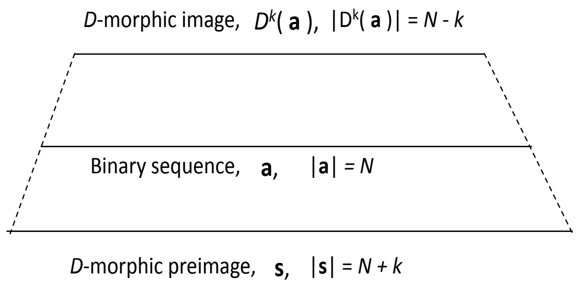
D-morphic image, $D^k(\mathbf{a})$, $|D^k(\mathbf{a})| = N - k$
Binary sequence, **a**, $|\mathbf{a}| = N$
D-morphic preimage, **s**, $|\mathbf{s}| = N + k$

Fig. 1: D-morphic image and preimage of a binary sequence

Definition 3. *A segment* $\boldsymbol{s}$ *of length* $(2n + k)$ *of a composited de Bruijn sequence of period* 2^{n+k} *is called a* k-th order D-morphic order n m-sequence preimage *((n, k)-DMMP) if* $D^k(\boldsymbol{s})$ *is a segment of length* $2n$ *of an* m*-sequence.*

Definition 4. *A de Bruijn sequence generated by recurrence relation (1) is referred to as a* linearly generated de Bruijn sequence *when the feedback function* g *is corresponding to a primitive polynomial.*

Definition 5. *A de Bruijn sequence generated by recurrence relation (1) is referred to as a* nonlinearly generated de Bruijn sequence *when the feedback function* g *is nonlinear.*

Enumeration of (n, k)-DMDPs. We use the higher order D-morphic image and preimages to analyze the composited de Bruijn sequences generated by recurrence relation (1). From now on, we simplify the term using (n, k)-DMDP instead of k-th order D-morphic order n de Bruijn preimage and (n, k)-DMMP instead of k-th order D-morphic order n m-sequence preimage. Since a de Bruijn sequence of order $(n + k)$ is produced by computing the D-morphic preimages, there exist many (n, k)-DMDPs in a composited de Bruijn sequence, as shown in Fig. 2. Below we calculate the number of (n, k)-DMDPs in a de Bruijn sequence of period 2^{n+k}.

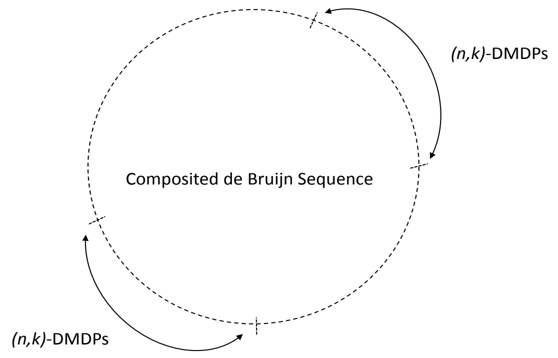

Fig. 2: Existence of (n, k)-DMDPs in a composited de Bruijn sequence

Lemma 2. *For a composited de Bruijn sequence of period* $2^{n+1}, n \geq 2$*, there exist exactly* $2n$ $(n, 1)$*-DMDPs.*

Proof. Let $\mathbf{s}$ be a composited de Bruijn sequence of period 2^{n+1} and $\mathbf{a} = \{a_i\}_{i \geq 0}$ be a starting de Bruijn sequence of period 2^n. Denote by $z = \{z_i\} = s_1 e s_2$ and $\bar{z} = \{\bar{z}_i\} = s_3 \hat{e} s_4$ two D-

morphic preimages of $\mathbf{a}$, where $z_0 = 0$, $z_i = \sum_{j=0}^{i-1} a_j$ and $\bar{z}_i = 1+z_i$, e and $\hat{e}$ are the conjugate pair. Then, $\mathbf{s} = s_1 e s_4 s_3 \hat{e} s_2$, where two bits of the conjugate pair overlap two preimages.

The segment $e_1 s_4 s_3 \hat{e} = 1010...10 \| s_4 s_3 \hat{e} = 1010...10 \| E^t(\bar{z})$ of length $(2^n + n)$, is a concatenation of e_1 and a one-order preimage of $\mathbf{a}$, and contains n $(n, 1)$-DMDPs, where e_1 is obtained by removing the first bit from e. Similarly, the segment $\hat{e_1} s_2 s_1 e = 1010...10 \| s_2 s_1 e = 1010...10 \| E^t(z)$ contains n $(n, 1)$-DMDPs, where $\hat{e_1}$ is obtained by removing the first bit from $\hat{e}$. No other segment of length $(2^n + 1)$ is an $(n, 1)$-DMDP in the de Bruijn sequence of period 2^{n+1}, as follows from the construction of $\mathbf{s}$. Therefore, the total number of $(n, 1)$-DMDPs in the de Bruijn sequence of period 2^{n+1} is equal to $2n$. Hence, the assertion is established. □

Theorem 2. *For any de Bruijn sequence of order $(n+k)$ generated by recurrence relation (1), the number of (n, k)-DMDPs is equal to $2n$.*

Proof. By Lemma 2, the de Bruijn sequence of period 2^{n+k} contains $2(n+k-1)$ $(n+k-1, 1)$-DMDPs. Similarly, the de Bruijn sequence of period 2^{n+k-1} contains $2(n+k-2)$ $(n+k-2, 1)$-DMDPs. As the de Bruijn sequence of period 2^{n+k-1} is constructed from the de Bruijn sequence of period 2^{n+k-2}, there cannot be more than $2(n+k-2)$ $(n+k-2, 2)$-DMDPs in the de Bruijn sequence of period 2^{n+k}. Applying the similar argument, in general, it follows that there are $2(n+k-i)$ $(n+k-i, i)$-DMDPs in the de Bruijn sequence of period 2^{n+k} for $i \leq k$. Therefore, for $i = k$, there are exactly $2n$ (n, k)-DMDPs in the de Bruijn sequence of period 2^{n+k}. □

Remark 3. The k-th order D-morphic images of (n, k)-DMDPs are de Bruijn sequences, which are shift equivalent to the starting de Bruijn sequence of period 2^n. Note that only $2n$ (n, k)-DMDPs out of 2^k (n, k)-DMDPs exist in a composited de Bruijn sequence.

How an (n, k)-DMDP is Vulnerable. We focus on the reconstruction of a composited de Bruijn sequence from a segment of the de Bruijn sequence. The reconstruction can be accomplished in two ways. The first strategy is to obtain an LFSR by the Berlekamp-Massey algorithm [25], which can generate the de Bruijn sequence. According to Theorem 1, the linear complexity of a composited de Bruijn sequence is high, and the LFSR reconstruction demands a huge bit stream and the length of the LFSR will be large. Therefore, we don't apply the Berlekamp-Massey algorithm to reconstruct the sequence. The second strategy is to recover the nonlinear feedback function of a composited de Bruijn sequence and then using $(n + k)$ bits as an internal state of the NLFSR the whole sequence can be reconstructed. We observe that an (n, k)-DMDP is a vulnerable segment and it allows one to

1. construct the starting de Bruijn sequence by applying k-th order D-morphism D^k on an (n, k)-DMDP and its nonlinear feedback function $h = (g + J^{n-1})$ by the Reed-Muller transformation with complexity $O(2^n)$.
2. first recover the feedback function of recurrence relation (1) by using h and the assumption in Section 4.2 and then reconstruct the composited de Bruijn sequence using any $(n + k)$ bits from an (n, k)-DMDP/DMMP as an initial state of the NLFSR. Symbolically,

$$R_k^n = \overbrace{(x_n + x_0) \circ \psi^k + G(x_1, x_2, ..., x_{n-1}) \circ \psi^k}^{\text{Recover from an } (n,k)\text{-DMDP/DMMP}} + \underbrace{I_k^n(x_1, ..., x_{n+k-1})}_{\text{Recover from } X_0^n \text{ and } J^{n-1}} .$$

3. reconstruct the composited de Bruijn sequence of period 2^{n+k} using the starting de Bruijn sequence by the algorithms in [1, 6].

We discuss the method of recovering the feedback function of recurrence relation (1) as well as the composited de Bruijn sequence and calculate the success probability of reconstructing a composited de Bruijn sequence in the following two subsections.

4.2 Reconstruction of a Linearly Generated de Bruijn Sequence

Although the recurrence relation (1) is nonlinear when g is a linear feedback function of a primitive polynomial, the de Bruijn sequences generated by such recurrence relations are not strong under the following assumption.

1. We assume that an attacker knows that a de Bruijn sequence is generated by the composited construction using recurrence relation (1), and knows parameters n and k, but the attacker does not know the feedback function. Using n and k and X_0^n and J^{n-1}, the attack can construct the product-of-sum terms and hence the feedback function I_k^n.
2. We also assume that an (n,k)-DMMP/(n,k)-DMDP of a de Bruijn sequence of period 2^{n+k} is known to the attacker.

In this scenario, the attacker's goal is to recover the linear feedback function g from a segment of the de Bruijn sequence. The method of recovering the feedback function g is described below.

```
**Method 1**: Recover the feedback function $g$
**Input:** An $(n,k)$-DMMP $S$ of length $(2n+k)$ of a de Bruijn sequence of period $2^{n+k}$.
**Output:** The feedback function $g$.
1: Calculate the $k$-th order $D$-morphism of $S$, i.e., $w = D^k(S)$.
2: Apply the Berlekamp-Massey algorithm on $w$ to recover the feedback polynomial
   or form a system of linear equations with $n$ variables and $n$ equations using $w$ and
   solve it to recover the feedback polynomial.
```

When an (n,k)-DMMP is known, applying the above method the attacker can obtain the feedback function g and hence the recurrence relation (1) as the function I_k^n can be formulated by the attacker from his knowledge of the construction. Upon recovery of the feedback function, using any $(n+k)$ consecutive bits as an internal state to the NLFSR, the composited de Bruijn sequence can be generated. However, any segment of length $(2n+k)$ cannot be used to recover the feedback function g, only the (n,k)-DMMPs allow one to recover the feedback function g. In the following theorem, we calculate the success probability of obtaining an (n,k)-DMMP of length $(2n+k)$ from a linearly generated de Bruijn sequence.

Theorem 3. *For any linearly generated de Bruijn sequence, the success probability of obtaining an (n,k)-DMMP is at least $\frac{2n(2^n-n)}{2^{n+k}} \approx \frac{n}{2^{k-1}}$ when $2^{k-1} \geq n$.*

Proof. By Theorem 2, there are exactly $2n$ (n,k)-DMDPs. As the starting de Bruijn sequence of period 2^n is produced from an m-sequence by inserting one zero to the run of zeros of length $(n-1)$, at least $(2^n - n)$ segments of length $2n$ allow one to recover the linear feedback polynomial and only n segments of length $2n$ that contain the zero tuple cannot determine the feedback function. Thus the number of (n,k)-DMMPs of length $(2n+k)$ in the de Bruijn sequence of period 2^{n+k} is at least $2n(2^n-n)$. Since the total number of segments of length $(2n+k)$ in the de Bruijn sequence of period 2^{n+k} is 2^{n+k}, the success probability of randomly obtaining an (n,k)-DMMP is at least $\frac{2n(2^n-n)}{2^{n+k}} \approx \frac{n}{2^{k-1}}$. □

4.3 Reconstruction of a Nonlinearly Generated de Bruijn Sequence

In this subsection, we calculate the success probability of obtaining an (n,k)-DMDP for a nonlinearly generated composited de Bruijn sequence. We also investigate the hardness of producing a composited de Bruijn sequence of period 2^{n+k} from a de Bruijn sequence of period 2^n and vice versa.

Contraction and Expansion of Composited de Bruijn Sequences. We now describe two naive approaches for producing a de Bruijn sequence of period 2^n from a de Bruijn sequence of period 2^{n+k} and vice versa.

The starting de Bruijn sequence of period 2^n can be produced from the de Bruijn sequence of period 2^{n+k} by repeatedly determining a de Bruijn sequence of order $(n+i-1)$ from a de Bruijn sequence of order $(n+i)$ or by finding an (n,k)-DMDP. In order to produce a de Bruijn sequence of lower order, one needs to know at least half of the sequence and one of the conjugate pair, and then needs to find the D-morphic image. Again, the task of producing a de Bruijn sequence of order n from a de Bruijn sequence of order $(n+k)$ is equivalent to the task of approximating the function I_k^n by J_k^{n-1} in recurrence relation (1).

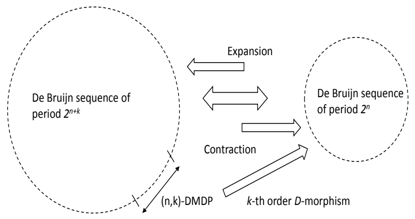
Expansion
De Bruijn sequence of period 2^{n+k}
De Bruijn sequence of period 2^n
Contraction
(n,k)-DMDP
k-th order D-morphism

Fig. 3: Contraction and expansion of de Bruijn sequences

Proposition 2. *Using the above contraction method, a de Bruijn sequence of oder n cannot be obtained from a de Bruijn sequence of order $(n+k)$ in less than $2^n(2^k-1)$ steps.*

Proof. As shown in Lemma 2 that a de Bruijn sequence of period 2^{n+k} can be written as $\mathbf{s} = s_1 e s_4 s_3 \hat{e} s_2$. The de Bruijn sequence $\mathbf{a}$ can be obtained by first finding either an $(n+k-1,1)$-DMDP or one of the conjugate pair (as es_4s_3 and $\hat{e}s_2s_1$ contain $(n+k-1,1)$-DMDPs) and then applying one order D-morphism on it. To find a segment of length $(2^{n+k-1}+1)$, i.e, an $(n+k-1,1)$-DMDP, the upper and lower bounds of the number of steps are 2^{n+k-1} and $(2^{n+k-1}+2^{n+k-1}) = 2^{n+k}$, respectively and the exact number of steps depends on the sequence $\mathbf{a}$ and the position of one of the conjugate pair. Hence, the number of steps required to produce a de Bruijn sequence of period 2^n from a de Bruijn sequence of period 2^{n+k} is at least $(2^{n+k-1}+2^{n+k-2}+\cdots+2^n) = 2^n(2^k-1)$. □

On the other hand, a de Bruijn sequence of order n can be expanded to a de Bruijn sequence of order $(n+k)$. Without using recurrence relation (1), the method of generating of a de Bruijn sequence of order $(n+k)$ from a de Bruijn sequence of order n can be found in [1, 6]. However, for large values of $(n+k)$ (e.g. $(n+k)=64$), this method cannot be applied due to data and memory complexities.

The Success Probability of Obtaining an (n,k)-DMDP. We now determine the success probability of obtaining an (n,k)-DMDP in the following theorem.

Theorem 4. *Let the linear complexity of a span n sequence generated by g in recurrence relation (1) be optimal. For any de Bruijn sequence of order $(n+k)$, the success probability of obtaining an (n,k)-DMDP is $\frac{n}{2^{n+k-1}}$.*

Proof. As the linear complexity of the span n sequence generated by g is optimal, the span n sequence cannot be generated by an LFSR of length n and one requires 2^n-2 bits to produce the span n sequence. According to Theorem 2, there are exactly $2n$ (n,k)-DMDPs in the de Bruijn sequence of period 2^{n+k}. Since the number of segments of length (2^n+k) in a de Bruijn sequence of period 2^{n+k} is 2^{n+k}, the success probability of obtaining an (n,k)-DMDP is equal to $\frac{2n}{2^{n+k}} = \frac{n}{2^{n+k-1}}$. □

Under the assumption in Section 4.2, by first finding an (n,k)-DMDP and then applying the Reed-Muller transformation on the k-th order D-morphic image of the (n,k)-DMDP, the nonlinear feedback function g can be recovered with complexity $O(2^n)$ and hence the feedback function for the recurrence relation (1). Once the feedback function for recurrence relation (1) is recovered, taking any $(n+k)$ consecutive bits from the (n,k)-DMDP as an internal state, the composited de Bruijn sequence can be reconstructed. Thus, an (n,k)-DMDP of a composited de Bruijn sequence allows one to reconstruct the de Bruijn sequence of period 2^{n+k}.

We note that the Berlekamp-Massey algorithm [25] can also be used for both linearly and nonlinearly generated de Bruijn sequences to reconstruct the composited de Bruijn sequence, but such reconstruction demands a huge number of bits due to the high linear complexity of a composited de Bruijn sequence (see Theorem 1). Again, the complexity of recovering the feedback function of recurrence relation (1) by the Reed-Muller transformation is at least $O(2^{n+k})$. Under the assumption stated in Section 4.2, the composited de Bruijn sequence can be reconstructed with complexity $O(2^n)$ using an (n,k)-DMDP, where the sequence generation cost is not included. For the linear case, the complexity of recovering the feedback function g is only (n^2) when an (n,k)-DMMP is known and the success probability of obtaining an (n,k)-DMMP is high. Contrariwise, for the nonlinear case, the complexity for recovering the feedback function is high and the success probability of obtaining an (n,k)-DMMP is low.

Remark 4. A direct LFSR approximation of the nonlinear feedback function g cannot be used in recurrence relation (1) as doing so would increase the number of stages of the NLFSR and the conjugate pairs are different for the approximated LFSR. In general, one needs to find a segment of length $(2L+k)$ or (2^n+k) whose k-th order D-morphic image is a segment of length $2L$ of a span n sequence to recover the nonlinear feedback function g, where L is the linear complexity of a span n sequence generated by g.

Selection Criteria for the Parameters. Based on the above analysis, we have the following criteria for the selection of parameters for generating strong de Bruijn sequences by the composited construction:

1. The value of n in recurrence relation (1) should be large, as it determines the complexity of finding an (n,k)-DMDP/(n,k)-DMMP and of recovering the function g, and it will reduce the number of compositions for a fixed value of $(n+k)$.
2. The function g must be a nonlinear function. Additionally, the linear complexity of the starting span n sequence generated by g should be optimal or near-optimal since an attacker will require more bits and have low probability of obtaining an (n,k)-DMDP of length (2^n+k) and the complexity to recover g will be high.

3. The value of k should be large subject to the condition of an efficient implementation (see Section 5.2) as a large value will minimize the success probability of obtaining an (n, k)-DMDP/(n, k)-DMMP according to Theorems 3 and 4.

5 Iterative Computation of Feedback Function I_k^n

In [24], an algebraic expression for each product-of-sum term of I_{16}^n is derived. Since I_k^n contains $(k+1)$ product-of-sum terms, it may not be efficient to implement all the product-of-sum terms of I_k^n in hardware for a large value of k. In this section, we present a new iterative method for computing I_k^n, which takes k iterations. We also propose an iterative parallel computation technique for I_k^n in which the number of iterations can be reduced.

5.1 Iterative Computation of X_{k+1}^p

We rewrite X_0^p as $X_0^p = \prod_{i=1}^{p} U_i$ where $U_{2j} = x_{2j} + 1, j = 1, 2, ...$ and $U_{2j+1} = x_{2j+1}, j = 0, 1, ...$ For $k \geq 1$, the $(k+1)$-th order composition of U_i can be written as $U_i \circ \psi^{k+1} = U_i \circ \psi^k + U_{i+1} \circ \psi^k$. Denote $X_k^p = \prod_{i=1}^{p} U_i \circ \psi^k = \prod_{i=1}^{p} V_i = V_1 \cdot V_2 \cdot V_3 \cdots V_{p-1} \cdot V_p$, $V_i = U_i \circ \psi^k$. Then,

$$\begin{aligned} X_{k+1}^p &= \prod_{i=1}^{p} U_i \circ \psi^{k+1} = \prod_{i=1}^{p} (U_i \circ \psi^k + U_{i+1} \circ \psi^k) \\ &= (V_1 + V_2 + 1) \cdot (V_2 + V_3) \cdot (V_3 + V_4 + 1) \cdots (V_p + x_{p+1} \circ \psi^k) \text{ if } p \text{ is odd} \\ &= (V_1 + V_2 + 1) \cdot (V_2 + V_3) \cdot (V_3 + V_4 + 1) \cdots (V_p + 1 + x_{p+1} \circ \psi^k) \text{ if } p \text{ is even.} \end{aligned}$$

Thus X_{k+1}^p can be computed from X_k^p by the above relation in which one requires to compute $(x_{p+1} \circ \psi^k)$ in each iteration. Consider each product-of-sum term X_j^p as a register of length p where the i-th cell of the shift register contains the value of V_i. An overview of the computation of X_{k+1}^p from X_k^n is depicted in Fig. 4. An iterative computation of the product-of-sum term X_{i+1}^p shows that one requires a p-stage register, however, all the product-of-sum terms can be computed using only one $(n+k-1)$-stage register. We describe the method in the next subsection.

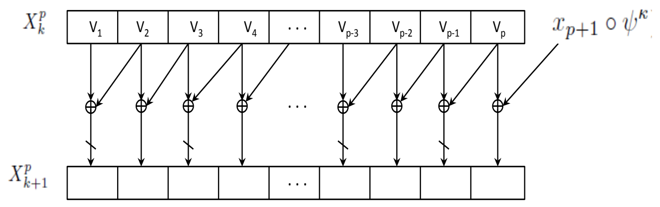

Fig. 4: An iterative computation of X_k^p

5.2 Iterative Computation and Complexity Estimation of I_k^n

This section presents the iterative technique for computing I_k^n, and gives an estimation for the number of XOR, multiplication and negation operations.

Iterative Computation of Function I_k^n. Without computing $(x_{p+1} \circ \psi^k)$ in X_{k+1}^p, the product-of-sum term X_{k+1}^p can be evaluated by using the result of $(p+1)$-th cell of X_k^{p+1} and the result is the same as $(x_{p+1} \circ \psi^k)$. In order to compute I_k^n, we need to compute each $X_j^p, j+p = (n+k-1), n \leq p \leq n+k-1$ and J_k^{n-1}. We use a register of $(n+k-1)$-bit for the product-of-sum term X_0^{n+k-1}. The initial value of the register is loaded as shown in Fig. 5. The product of first $(n+k-1-j)$ cells of the register gives the value of $X_j^{n+k-1-j}$. The iterative computation of X_0^{n+k-1} gives the results of all the product-of-sum terms where one product-of-sum term is evaluated, as shown in Section 5.1, in an iteration. Note that, at the first iteration, the content of $(n+k-1)$-th cell is not updated as the content of $(n+k-1)$-th cell is not required while computing X_1^{n+k-2}. Similarly, the contains of last j cells are not updated as $X_j^{n+k-1-j}$ is the product of the contents of first $(n+k-1-j)$ cells of the register. The iterative computation of the k-th order composition of X_0^{n+k-1} is illustrated in Fig. 5. Thus, by employing a register of length $(n+k-1)$, one can compute all the product-of-sum terms X_j^p in $(k-1)$ iterations. For $k = 2^l$, J_k^{n-1} can be computed in one iteration as $x_i \circ \psi^k = x_i + x_{i+k}$. For $k \neq 2^l$, J_k^{n-1} can be computed by employing another register of length $(n+k-1)$ in k iterations while the initial state of the register is the same as the content of the NLFSR and during performing multiplication the contents of the register are complemented. Therefore, the function I_k^n can be computed in k iterations. Table 1 exhibits a comparison of the storage requirement and time required to compute one bit for the iterative technique and other de Bruijn sequence generation techniques.

Table 1: The time and storage requirement comparison

Methods	Storage	Time required to produce one bit
Our Iterative Technique	$3(n+k)-2$	k
Fredricksen's Algorithm [11]	$3(n+k)$	about $(n+k)$
Jansen *et al.*'s Cross-joining Method [20]	$3(n+k)$	$4(n+k)$

Estimation of the Number of Different Operations. We have seen that the function I_k^n can be computed in an iterative way using XOR, multiplication and negation operations over $\mathbb{F}_2$, where each iteration computes a product-of-sum term. We give an estimation of the number of XOR, multiplication and negation operations required to implement the function I_k^n. According to the iterative formula in Section 5.1 for X_j^{n+k-1}, we require $(n+k-2)$ XOR and $\lceil \frac{(n+k-1)}{2} \rceil$ negation operations. On the other hand, the number of multiplication operations required for computing X_0^{n+k-1} is $(n+k-2)$ as X_0^{n+k-1} is a product of $(n+k-1)$ terms. Again, the number of XOR and multiplication operations for J_k^{n-1} is equal to $(n+k-2)$ and $(n-2)$, respectively for $k \neq 2^l$ and $l \geq 0$. Thus, the total number of XORs required to implement I_k^n is equal to $2(n+k-2)$ and the total number of multiplication operation required to implement I_k^n is equal to $(2n+k-4)$. We summarize the number of different operations required for implementing I_k^n in Table 2. The number of XOR operations can be reduced by half by choosing $k = 2^l$ for $l \geq 0$.

5.3 Parallel Iterative Computation of Function I_k^n

We now focus on how to reduce the number of iterations required for computing I_k^n. The number of iterations can be reduced by half if two registers of length $(n+k-1)$ are employed for computing $X_i^p, (i+p) = (n+k-1), 0 \leq i \leq k-1$. One register is used to compute

Table 2: Number of operations for I_k^n

	Operations		
	XOR	Multiplication	Negation
X_i^j	$(n+k-2)$	$(n+k-2)$	$\lceil\frac{(n+k-1)}{2}\rceil$
J_k^{n-1}	$(n+k-2)$	$(n-2)$	$(n-1)$

all X_j^{n+k-1}'s for odd values of j while the other one is used to compute all X_j^p's for even values of j. The iterative procedure for computing X_{2i}^p and X_{2i+1}^p is described in Fig. 6. For computing all odd order compositions X_{2i+1}^p, the initial state of the first register is loaded as

$$\begin{aligned} U_i &= x_i, \text{ if } i \text{ is odd}, 1 \le i \le (n+k-1) \\ &= x_i + 1, \text{ if } i \text{ is even}, 1 \le i \le (n+k-1). \end{aligned}$$

On the other hand, for computing all even order compositions X_{2i}^p, the initial state of the second register is loaded as

$$\begin{aligned} U_i &= x_i + x_{i+1}, \text{ if } i \text{ is odd}, 1 \le i \le (n+k-1) \\ &= x_i + x_{i+1} + 1, \text{ if } i \text{ is even}, 1 \le i \le (n+k-1). \end{aligned}$$

Therefore, in $\lceil\frac{k}{2}\rceil$ iterations, the function I_k^n can be evaluated using two registers of length $(n+k-1)$. Similarly, using four registers of length $(n+k-1)$, the function I_k^n can be computed

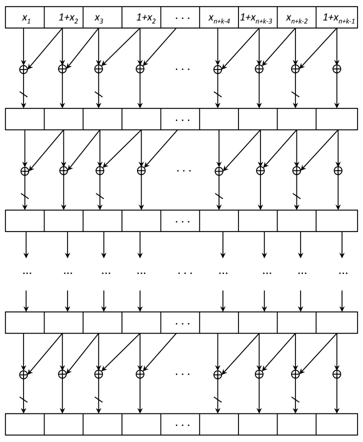

Fig. 5: Iterative computation of I_k^n

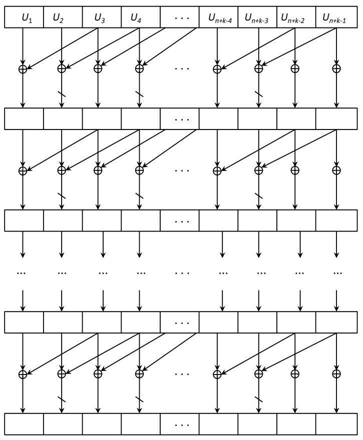

Fig. 6: Computation of X_{2i}^p and X_{2i+1}^p

in $\lceil\frac{k}{4}\rceil$ iterations where the i-th register computes the terms $X_{4j+i}^p, 0 \le i \le 3, j = 0, 1, ...$ In general, by employing more registers, the function I_k^n can be computed in a small number of iterations. However, the employment of more registers increases the hardware complexity.

6 Implementation of Three 64-stage NLFSRs

In this section, we present algebraic forms of three NLFSRs that generate de Bruijn sequences with period 2^{64}, and show their performances when they are implemented in software. Three 64-stage NLFSRs are determined by choosing $n = 23$ and $k = 41$, $n = 24$ and $k = 40$, and

$n = 27$ and $k = 37$ where each n-stage NLFSR generates a span n sequence. As far as we are concerned, these three NLFSRs generate the longest de Bruijn sequences whose algebraic forms are known. In [24], eight NLFSRs that generate de Bruijn sequences with periods in the range of 2^{35} and 2^{40} are presented.

6.1 Mathematical Details of Three 64-stage NLFSRs

The n-stage NLFSR for generating a span n sequence is defined by

$$\begin{aligned} x_{n+i} &= x_i + f(x_{r_1+i}, x_{r_2+i}, ..., x_{r_{2t}+i}) \\ &= x_i + WG1(x_{r_1+i}, ..., x_{r_t+i}) + WG2(x_{r_{t+1}+i}, ..., x_{r_{2t}+i}) \end{aligned} \tag{2}$$

where f is the sum of two different WG transformations over $\mathbb{F}_{2^t}$ and $0 < r_1 < r_2 \cdots < r_{2t} < n$. A graphical representation of the recurrence relation is provided in Fig. 7. For any choice of a $2t$-tap position $(r_1, r_2, ..., r_{2t})$ and WG transformations, the above recurrence relation may not generate a span n sequence. The reason for choosing the above type of recurrence relation is to take a simple feedback function with small number of monomials in the recurrence relation. Table 3 contains two span n sequences for $t = 5$, which are used to construct first two de Bruijn sequences. Moreover, the linear complexity of a span n sequence produced by the above recurrence relation for f_1 is optimal, i.e, $2^{23} - 2$. Fig. 8 depicts a general architecture of composited de Bruijn sequences using two WG transformations. When the recurrence relation for a span n sequence is not the sum of two WG transformations, two WG transformations are replaced by a function f that generates a span n sequence in Fig. 8. We present more de Bruijn sequences of period 2^{64} in Table 5 in Appendix A.

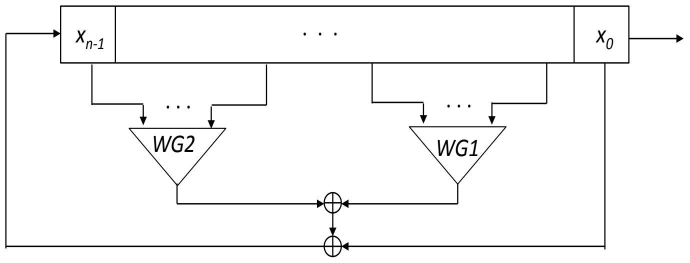

Fig. 7: A block diagram for generating span n sequences using two WG trans.

Table 3: WG transformation modules for span n sequences

	$WG1$			$WG2$			
	$(r_1, r_2, r_3, r_4, r_5)$	$(c_0, c_1, c_2, c_3, c_4)$	d	$(r_6, r_7, r_8, r_9, r_{10})$	$(c_0, c_1, c_2, c_3, c_4)$	d	n
f_1	$(1, 2, 3, 4, 5)$	$(1, 0, 0, 1, 0)$	1	$(7, 10, 12, 13, 20)$	$(1, 1, 1, 1, 0)$	15	23
f_2	$(1, 2, 3, 4, 6)$	$(1, 0, 1, 0, 0)$	15	$(10, 13, 15, 17, 21)$	$(1, 1, 0, 1, 1)$	1	24

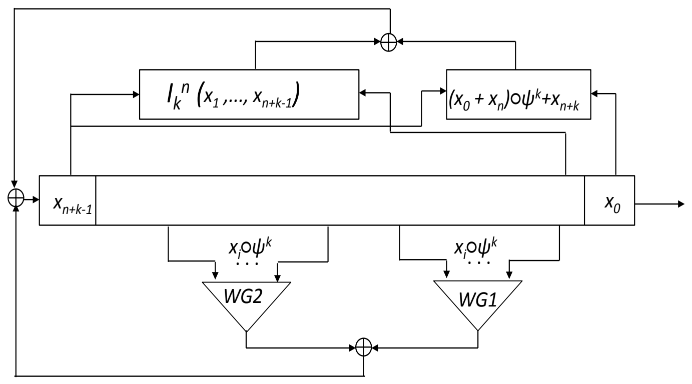

Fig. 8: A block diagram of composited de Bruijn sequences using WG trans.

The 64-stage NLFSR for $n = 23$ and $k = 41$ (NLFSR1) The explicit form of the 64-stage NLFSR for $t = 5$ is given by

$$\begin{aligned}x_{64} = {} & x_0 + x_1 + x_8 + x_9 + x_{32} + x_{33} + x_{40} + x_{41} + x_{23} + x_{24} + x_{31} + x_{32} + x_{55} + x_{56} \\ & + x_{63} + f_1(x_{r_1} + x_{r_1+1} + x_{r_1+8} + x_{r_1+9} + x_{r_1+32} + x_{r_1+33} + x_{r_1+40} + x_{r_1+41}, \\ & \ldots, x_{r_{2t}} + x_{r_{2t}+1} + x_{r_{2t}+8} + x_{r_{2t}+9} + x_{r_{2t}+32} + x_{r_{2t}+33} + x_{r_{2t}+40} + x_{r_{2t}+41}) + I_{41}^{23}\end{aligned}$$

where $J_{41}^{22} = \prod_{i=1}^{22}(x_i + x_{i+1} + x_{i+8} + x_{i+9} + x_{i+32} + x_{i+33} + x_{i+40} + x_{i+41})$.

The 64-stage NLFSR for $n = 24$ and $k = 40$ (NLFSR2) The explicit form of the 64-stage NLFSR for $t = 5$ is given by

$$\begin{aligned}x_{64} = {} & x_0 + x_8 + x_{32} + x_{40} + x_{24} + x_{32} + x_{56} + f_2(x_{r_1} + x_{r_1+8} + x_{r_1+32} + x_{r_1+40}, \\ & \ldots, x_{r_{2t}} + x_{r_{2t}+8} + x_{r_{2t}+32} + x_{r_{2t}+40}) + I_{40}^{24}\end{aligned}$$

where $J_{40}^{23} = \prod_{i=1}^{23}(x_i + x_{i+8} + x_{i+32} + x_{i+40})$.

The 64-stage NLFSR for $n = 27$ and $k = 37$ (NLFSR3) The explicit form of the 64-stage NLFSR is given by

$$\begin{aligned}x_{64} = {} & x_0 + x_1 + x_4 + x_5 + x_{32} + x_{33} + x_{36} + x_{37} + x_{27} + x_{28} + x_{31} + x_{32} + x_{59} + x_{60} \\ & + x_{63} + f_3(x_{r_1} + x_{r_1+1} + x_{r_1+4} + x_{r_1+5} + x_{r_1+32} + x_{r_1+33} + x_{r_1+36} + x_{r_1+37}, \\ & \ldots, x_{r_{2t}} + x_{r_{2t}+1} + x_{r_{2t}+4} + x_{r_{2t}+5} + x_{r_{2t}+32} + x_{r_{2t}+33} + x_{r_{2t}+36} + x_{r_{2t}+37}) + I_{37}^{27}\end{aligned}$$

where $J_{37}^{26} = \prod_{i=1}^{26}(x_i+x_{i+1}+x_{i+4}+x_{i+5}+x_{i+32}+x_{i+33}+x_{i+36}+x_{i+37})$ and $f_3(x_1,\ldots,x_{26}) = x_4+x_8+x_9+x_{11}+x_{12}+x_{15}+x_{16}+x_{23}+x_{12}x_{22}+x_{13}x_{23}+x_{13}x_{25}+x_{22}x_{23}+x_7x_8x_{24}+x_{12}x_{14}x_{26}+x_6x_{11}x_{19}x_{22}$, which is taken from [28] and is not the sum of two WG transformations.

6.2 Software Implementation

We implemented the above three NLFSRs in C (compiled with gcc 4.4.7 (for server) and gcc 4.6.3 (for desktop PC)) on a 2.13 GHz Intel(R) Xeon(R) CPU E7-L8867 server with 80 cores using 500 GB RAM and on a 2.66 GHz Intel(R) Core(TM)2 Quad CPU Q8400 desktop with 8 GB RAM. For optimizing the computation of the WG transformations, we use the algebraic normal form (ANF) representation for a WG transformation as it contains a small number of terms. We also implemented NLFSR1 and NLFSR2 using the look up tables for the WG transformations, but the throughput is lower than the throughput of the implementation using ANF representations. We evaluated the product-of-sum terms $X_i^j, (i+j) = 64, 0 \leq i \leq (k-1)$ in the function I_k^n by the newly proposed iterative method. Moreover, the sum terms $x_i \circ \psi^k$'s for $i = 0, n$ and $2t$ tap positions are expanded in the implementation for $k = 37, 40$, and 41. For $k = 37, 40$ and 41, the expansion of $x_i \circ \psi^k$ contains 8, 4 and 8 variables, respectively. Table 4 indicates the performance of three NLFSRs on the server as well as on the desktop where the performance is measured by the number of bits produced by an NLFSR per second.

Table 4: Performance of the NLFSRs in Kilobits per second (Kbps)

NLFSRs	Parameters	Kbps (Server)	Kbps (Desktop)
NLFSR1	$n = 23, k = 41$	78.79	80.88
NLFSR2	$n = 24, k = 40$	81.00	81.52
NLFSR3	$n = 27, k = 37$	83.28	85.13

In order to compare the efficiency, we also implemented a simple 64-stage LFSR with six tap positions, and an 64-stage NLFSR with six tap positions and a product term of 63 variables (J^{63}), where the NLFSR generates a de Bruijn sequence. The throughput of the LFSR and the NLFSR are 5.54 Megabits per second (Mbps) and 4.75 Mbps, respectively, both of which are much larger than the throughputs of the above three NLFSRs. However, the sequences generated by the LFSR and NLFSR are not secure because only $2 \times 64 = 128$ bits of the sequence are sufficient to produce the rest of the bits by recovering the characteristic polynomial. The de Bruijn sequences generated by NLFSR1, NLFSR2 and NLFSR3 are secure according to our analysis, especially, the de Bruijn sequences have high linear complexity.

7 Application of de Bruijn Sequences

A long de Bruijn sequence produced by the composited construction can be used as a random number generator where each number occurs exactly once in a period. An NLFSR that generates a long de Bruijn sequence is suitable for the digital signature generation algorithms where each random number should occur only once while generating digital signatures to protect the private key. A composited NLFSR can also be used as a building block for designing pseudorandom number generators and stream ciphers. In particular, the presented three NLFSRs are capable of generating all the numbers uniquely in the range of 0 and $2^{64}-1$. An NLFSRs that generates a de Bruijn sequence of period 2^{64} can be used as an alternative to a congruential generator for generating 64-bit random numbers on Linux operating system. To make use of the composited NLFSRs in passive RFID tags, the composited construction requires an efficient hardware implementation.

8 Conclusions

In this paper we first determined the linear complexity of a composited de Bruijn sequence and then analyze the recursive construction in a profound manner by introducing the notion of the higher order D-morphism and its preimages of a binary sequence. We calculated the number of (n, k)-DMDPs in a composited de Bruijn sequence and the success probability of obtaining an (n, k)-DMDP/(n, k)-DMMP which allows one to construct the starting de Bruijn sequence as well as its feedback function. The success probability of obtaining an (n, k)-DMDP is very low for a nonlinear feedback function. Our analysis shows that the de Bruijn sequences generated by the composition method with a nonlinear feedback function are strong and that can be used as building blocks for cryptographic applications. Furthermore, we presented a new iterative technique with its parallel extension for computing the product-of-sum terms of the feedback function. Due to the parallel computation, the number of iterations for evaluating the feedback function is minimized. In addition, we presented three NLFSRs that generate de Bruijn sequences of period 2^{64} and measured the performance of three NLFSRs when implementing the NLFSRs in software.

References

1. Annexstein, F.S.: Generating De Bruijn Sequences: An Efficient Implementation. IEEE Transactions on Computers 46(2), 198-200 (1997)
2. Berlekamp, E.R.: Algebraic Coding Theory, Ch. 7. McGraw-Hill, New York (1968)
3. Chan, A.H., Games, R.A.: On the Quadratic Spans of de Bruijn Sequences. IEEE Transactions on Information Theory 36(4), 822 – 829 (1990)
4. Chan, A.H., Games, R.A., Key, E.L.: On the Complexities of de Bruijn Sequences. Journal of Combinatorial Theory, Series A, 33(3), 233 – 246 (1982)
5. Chen, L., Gong, G.: Communication System Security. Chapman & Hall/CRC, Florida, USA (2012)
6. Chang, T., Park, B., Kim, Y. H., Song I.: An Efficient Implementation of the D-Homomorphism for Generation of de Bruijn Sequences. IEEE Transactions on Information Theory, 45(4), 1280 – 1283 (1999)
7. de Bruijn, N.G.: A Combinatorial Problem. Proc. Koninklijke Nederlandse Akademie v. Wetenschappen 49, 758 – 764 (1946)
8. The eStream Project. `http://www.ecrypt.eu.org/stream/`
9. Etzion, T., Lempel, A.: Construction of de Bruijn Sequences of Minimal Complexity. IEEE Transactions on Information Theory 30(5), 705 – 709 (1984)
10. Fredricksen, H.: A Survey of Full Length Nonlinear Shift Register Cycle Algorithms. SIAM Review 24(2), 195 – 221 (1982)
11. Fredricksen, H.: A Class of Nonlinear de Bruijn Cycles. Journal of Combinatorial Theory, Series A 19(2), 192 – 199 (1975)
12. Golomb, S.W.: On the Classification of Balanced Binary Sequences of Period $2^n - 1$. IEEE Transformation on Information Theory 26(6), 730 – 732 (1980)
13. Golomb, S. W.: Shift Register Sequences. Aegean Park Press, Laguna Hills, CA (1981)
14. Golomb, S. W., Gong, G.: Signal Design for Good Correlation: For Wireless Communication, Cryptography, and Radar. Cambridge University Press, New York (2004)
15. Gong, G., Youssef, A.: Cryptographic Properties of the Welch-Gong Transformation Sequence Generators. IEEE Transactions on Information Theory 48(11), 2837 – 2846 (2002)
16. Gong, G.: Randomness and Representation of Span n Sequences. In Proceedings of the 2007 International Conference on Sequences, Subsequences, and Consequences, SSC'07, pp. 192 – 203. Springer, Heidelberg (2007)
17. Good, I.J.: Normal Recurring Decimals. Journal of London Math. Soc. 21(3) (1946)
18. Green, D. H., Dimond, K. R.: Nonlinear Product-Feedback Shift Registers. In Proceedings of the Institution of Electrical Engineers, Vol. 117, Issue 4, pp. 681 – 686 (1970)
19. Green, D. H., Dimond, K. R.: Some Polynomial Compositions of Nonlinear Feedback Shift Registers and their Sequence-Domain Consequences. In Proceedings of the Institution of Electrical Engineers, Vol. 117, Issue 9, pp. 1750 – 1756 (1970)
20. Jansen, C.J.A., Franx, W.G., Boekee, D.E.: An Efficient Algorithm for the Generation of de Bruijn Cycles. IEEE Transactions on Information Theory 37(5), 1475 – 1478 (1991)

21. Kjeldsen, K.: On the Cycle Structure of a Set of Nonlinear Shift Registers with Symmetric Feedback Functions. Journal Combinatorial Theory Series A 20, 154 – 169 (1976)
22. Lempel, A.: On a Homomorphism of the de Bruijn Graph and its Applications to the Design of Feedback Shift Registers. IEEE Transactions on Computers C-19(12), 1204 – 1209 (1970)
23. Mandal, K., Gong, G.: Probabilistic Generation of Good Span n Sequences from Nonlinear Feedback Shift Registers. CACR Technical Report (2012)
24. Mandal, K., Gong, G.: Cryptographically Strong de Bruijn Sequences with Large Periods. In: Knudsen, L.R., Wu, H. (Eds.) SAC 2012. LNCS, vol. 7707, pp. 104 – 118. Springer, Heidelberg (2012)
25. Massey, J.L.: Shift-Register Synthesis and BCH Decoding. IEEE Trans. Inform. Theory, vol. 15, No. 1, 122–127 (1969)
26. Mayhew, G.L., Golomb, S.W.: Characterizations of Generators for Modified de Bruijn Sequences. Advanced Applied Mathematics 13(4), 454 – 461 (1992)
27. Mykkeltveit, J., Siu, M-Keung., Tong, P.: On the Cycle Structure of Some Nonlinear Shift Register Sequences. Information and Control 43(2), 202 – 215 (1979)
28. Rachwalik, T., Szmidt, J., Wicik, R., Zablocki, J.: Generation of Nonlinear Feedback Shift Registers with Special-Purpose Hardware. Report 2012/314, Cryptology ePrint Archive (2012), `http://eprint.iacr.org/`
29. `http://www.ecrypt.eu.org/stream/ciphers/achterbahn/achterbahn.pdf`

A. Appendix

In this appendix, we present some instances of composited de Bruijn sequences of period 2^{64}. In order to construct a de Bruijn sequence of period 2^{64}, we choose $n = 23$ and 24 and $k = 41$ and 40, respectively. For generating span n sequences for $n = 23$ and 24, we use the recurrence relation (2). Table 5 contains the span n sequences with periods $(2^{23} - 1)$ and $(2^{24} - 1)$. The first four functions in the table generate span n sequences of period $(2^{23} - 1)$, and the last two functions generate span n sequences of period $(2^{24} - 1)$.

Table 5: WG transformation modules for six span n sequences

	$WG1$			$WG2$		
	$(r_1, r_2, r_3, r_4, r_5)$	$(c_0, c_1, c_2, c_3, c_4)$	d	$(r_6, r_7, r_8, r_9, r_{10})$	$(c_0, c_1, c_2, c_3, c_4)$	d
f_3	$(1,2,3,4,7)$	$(1,0,1,0,0)$	15	$(8,12,14,16,20)$	$(1,0,0,1,0)$	11
f_4	$(1,2,3,5,7)$	$(1,0,1,0,0)$	11	$(8,9,13,14,18)$	$(1,1,1,1,0)$	1
f_5	$(1,2,3,5,12)$	$(1,1,1,0,1)$	15	$(13,15,17,20,22)$	$(1,1,0,1,1)$	15
f_6	$(1,2,3,6,7)$	$(1,1,1,0,1)$	3	$(10,15,16,18,20)$	$(1,1,1,1,0)$	1
f_7	$(1,2,3,4,11)$	$(1,0,0,1,0)$	7	$(14,16,18,21,22)$	$(1,0,0,1,0)$	1
f_8	$(1,3,4,5,6)$	$(1,0,1,1,1)$	7	$(10,11,12,20,23)$	$(1,1,0,1,1)$	11

The 64-stage NLFSR for $n = 23$ and $k = 41$ The explicit form of the 64-stage NLFSR for $t = 5$ is given by

$$\begin{aligned} x_{64} = {} & x_0 + x_1 + x_8 + x_9 + x_{32} + x_{33} + x_{40} + x_{41} + x_{23} + x_{24} + x_{31} + x_{32} + x_{55} + x_{56} \\ & + x_{63} + f_i(x_{r_1} + x_{r_1+1} + x_{r_1+8} + x_{r_1+9} + x_{r_1+32} + x_{r_1+33} + x_{r_1+40} + x_{r_1+41}, \\ & ..., x_{r_{2t}} + x_{r_{2t}+1} + x_{r_{2t}+8} + x_{r_{2t}+9} + x_{r_{2t}+32} + x_{r_{2t}+33} + x_{r_{2t}+40} + x_{r_{2t}+41}) + I_{41}^{23} \end{aligned}$$

where $J_{41}^{22}=\prod_{i=1}^{22}(x_i+x_{i+1}+x_{i+8}+x_{i+9}+x_{i+32}+x_{i+33}+x_{i+40}+x_{i+41})$, f_i provided in Table 5 is the sum of two different WG transformations for $3\leq i\leq 6$.

The 64-stage NLFSR for $n=24$ and $k=40$ The explicit form of the 64-stage NLFSR for $t=5$ is given by

$$\begin{aligned}x_{64}=x_0+x_8+x_{32}+x_{40}+x_{24}+x_{32}+x_{56}+f_j(x_{r_1}+x_{r_1+8}+x_{r_1+32}+x_{r_1+40},\\ ...,x_{r_{2t}}+x_{r_{2t}+8}+x_{r_{2t}+32}+x_{r_{2t}+40})+I_{40}^{24}\end{aligned}$$

where $J_{40}^{23}=\prod_{i=1}^{23}(x_i+x_{i+8}+x_{i+32}+x_{i+40})$, f_j, $j=7$ and 8 is the sum of two different WG transformations.