The Shortest Feedback Shift Register That Can Generate A Given Sequence

CEES J.A. JANSEN
Philips USFA B.V.
P.O. Box 218
5600 MD Eindhoven
The Netherlands

DICK E. BOEKEE
Technical University of Delft
P.O. Box 5031
2600 GA Delft
The Netherlands

Abstract

In this paper the problem of finding the absolutely shortest (possibly nonlinear) feedback shift register, which can generate a given sequence with characters from some arbitrary finite alphabet, is considered. To this end, a new complexity measure is defined, called the maximum order complexity. A new theory of the nonlinear feedback shift register is developed, concerning elementary complexity properties of transposed and reciprocal sequences, and feedback functions of the maximum order feedback shift register equivalent. Moreover, Blumer's algorithm is identified as a powerful tool for determining the maximum order complexity profile of sequences, as well as their period, in linear time and memory. The typical behaviour of the maximum order complexity profile is shown and the consequences for the analysis of given sequences and the synthesis of feedback shift registers are discussed.

1 Introduction

The vast majority of implemented cipher systems consists of streamcipher systems. In a streamcipher system each plaintext block is enciphered with a varying encipherment transformation, where the variation is on a block sequence base such as time or storage location. Therefore, identical plaintext blocks usually do not result in identical ciphertext blocks. In streamciphers the variation of the encipherment transformation inherently implies the presence of memory, whose internal state changes with every subsequent block according to some rule. Examples of streamciphers are the DES in any of its feedback modes [12,3], the *running key generator* (RKG) [8] and the *one-time pad* or *Vernam* cipher [3].

The running key generator is usually depicted as an autonomous finite state device that generates a sequence which is ultimately periodic. This sequence is then added to the stream of plaintext characters. It was the impracticability of the one-time-pad that led to streamciphers based on running key generators. The perfect

G. Brassard (Ed.): Advances in Cryptology - CRYPTO '89, LNCS 435, pp. 90-99, 1990.

secrecy of the one-time-pad [15] is approached by not using a random keystream, but rather a keystream generated by some finite state device, acting on a finite length, secret, randomly chosen key. This keystream, produced by a running key generator, should resemble a random keystream as much as possible. In particular, the unpredictability of successive keystream characters should be maintained as long as possible. It turns out that perfect statistical properties and unpredictability are not equivalent, the best example being sequences generated by linear feedback shift registers.

Many people have studied this, seemingly difficult, controversy. Well-known in this respect are Golomb's randomness postulates [5], which measure the randomness of a periodic binary sequence, viz. the disparity between ones and zeroes within one period, the run-length distribution and the number of values assumed by the periodic autocorrelation. Lempel and Ziv [9] introduced a complexity measure for finite sequences, based on the recursive copying of parts of a sequence. Rueppel [14] considered as a measure of randomness the so-called linear complexity profile, denoting the length of the shortest linear feedback shift register which generates that part of the sequence which has already been considered.

Elaborating on Rueppel's work we propose a complexity measure in this paper, called *maximum order complexity*, which denotes in a similar fashion the length of the shortest feedback shift register to generate a given (part of a) sequence, where the feedback function may be any function, mapping states onto characters. The name maximum order complexity was chosen because, unlike with linear (or first order) complexity, quadratic (or second order) complexity, etc., there is no restriction on the nonlinear order of the feedback function.

The import of maximum order complexity is that it tells exactly how many keystream characters have to be observed at least, in order to be able to generate the entire sequence by means of a feedback shift register of that length. Also maximum order complexity can be viewed as an additional figure of merit to judge the randomness of sequences.

2 Theory

In this section we present a summary of the theory of maximum order complexity. Proofs are omitted for the sake of brevity, but can be found in [6]. Section 2.1 introduces maximum order complexity and its elementary properties. In Section 2.2 an algorithm to determine the maximum order complexity profile is discussed. The typical behaviour of the maximum order complexity profile is shown in Section 2.3. Finally, Section 2.4 takes up the problem of the analysis of pseudo-random sequences and the resynthesis with feedback shift registers.

2.1 Maximum Order Complexity

Consider the following problem. Given a sequence $\underline{s} = (\alpha_0, \alpha_1, \ldots, \alpha_{l-1})$ of length l, with characters $\alpha_i \in \mathcal{A}$, where the alphabet $\mathcal{A}$ is some finite set. How many sections (i.e. memory cells) should a feedback shift register at least have in order to generate the sequence $\underline{s}$? So regardless of what the (memoryless) feedback function would have to be, linear or nonlinear. To this end, the following complexity measure is defined:

Definition 1 *The maximum order complexity $c(\underline{s})$ of a sequence $\underline{s} = (\alpha_0, \alpha_1, \ldots, \alpha_{l-1})$ with characters $\alpha_i \in \mathcal{A}$, where the alphabet $\mathcal{A}$ is some finite set, is defined to be the length L of the shortest feedback shift register for which there exists a memoryless feedback mapping, such that the FSR can generate the sequence $\underline{s}$.*

Maximum order complexity is expressed as being L characters. By this it is implicitly assumed that the memory cells can only contain characters from the alphabet $\mathcal{A}$.

Associated with any feedback function F is a substitution table or truth table, which can be seen as a list of argument values with the corresponding function values. The memory cells of the FSR provide for the argument values and hence the truth table is determined by the sequence $\underline{s}$. In general it is possible that a truth table is not specified completely by the sequence it generates, in which case there are no function values specified for one or more argument values.

Maximum order complexity has a number of basic properties, viz.:

Proposition 1

1. *For a sequence $\underline{s}$ consisting of two or more possibly repeated different characters, the complexity $c(\underline{s})$ is equal to the length-plus-one of the longest subsequence that occurs at least twice with different successor characters.*
2. *The complexity of a sequence is 0 iff this sequence consists of one possibly repeated character.*
3. *The maximum value of the complexity of a sequence of length l is $l-1$. A sequence of length l has a complexity of $l-1$ iff the sequence consists of $l-1$ consecutive copies of some character, followed by an unidentical character.*

Periodic sequences of period p are denoted by $\underline{s} = (\alpha_0, \alpha_1, \ldots, \alpha_{p-1})^\infty$. With the period we mean the least integer p, such that $\forall_{i \geq 0} [\alpha_{i+p} = \alpha_i]$. For periodic sequences we have the following property:

Proposition 2

1. *The minimum complexity of a periodic sequence of period p is $\lceil \log_a p \rceil$, where $a = |\mathcal{A}|$, the cardinality of the character alphabet.*
2. *The maximum complexity of a periodic sequence of period p is $p-1$.*

It seems natural for a FSR to also consider the complexity or degree of difficulty of the feedback function itself. One could consider for example the number of terms and highest degree in some representation of the function like the *Disjunctive Normal Form* (DNF, see e.g. [10, pg. 370]) or the *Algebraic Normal Form* (ANF, see e.g. [14, pg. 54]). As is the case with many complexity measures, the relation between high or low complexity and cryptographically good or bad sequences is not straightforward. Just as with linear complexity high maximum order complexity sequences are not necessarily cryptographically good, as demonstrated by the sequence $(00\cdots01)$ of length l and complexity $l-1$. Clearly, one has to find out the typical complexity values of good sequences or better even the typical complexity profile as done by Rueppel in [14, Ch. 4] for linear complexity.

From our definition of complexity it can be seen that in case the feedback function turns out to be linear, maximum order complexity is equal to linear complexity. This situation occurs with the so-called pseudo-noise or PN-sequences (sometimes called maximum-length or ML-sequences) of period 2^c-1, see e.g. [5].

Example 1 Consider the following sequence of length 25, obtained with an unbiased dice: $\underline{s}_D = (65445525664334341625314 33)$. It has a complexity of 3 characters, as all the subsequences of length 3 are distinct, but subsequence (43) has two different successors.

Consider again a sequence $\underline{s} = (\alpha_0, \alpha_1, \ldots, \alpha_{l-1})$, with $\alpha_i \in \mathcal{A}$. The transposed sequence $\underline{t} = T\underline{s} = (\beta_0, \beta_1, \ldots, \beta_{l-1})$, with $\beta_i \in \mathcal{B} = T_{\mathcal{A}}$ is defined to be the sequence which is obtained by substituting each character α_i of $\underline{s}$ by a character β_i from the alphabet $\mathcal{B}$, where the transposition operator T induces a one-to-one correspondence between the α_i and the β_i for all i, $0 \leq i \leq l-1$. For these transposed sequences we have the following result:

Proposition 3 *For all sequences $\underline{s}$ the maximum order complexity of $\underline{s}$ and that of its transpose $T\underline{s}$ have the same value.*

As a consequence, in the binary case the complementary sequence is generated by a feedback function which has inclusion and multiplication interchanged in the DNF representation.

Next we restrict ourselves to periodic sequences of period p. For this type of sequences we have the following result:

Proposition 4 *A periodic sequence $\underline{s} = (\alpha_0, \alpha_1, \ldots, \alpha_{p-1})^\infty$ and its reciprocal $\underline{s}^* = (\alpha_{p-1}, \ldots, \alpha_1, \alpha_0)^\infty$ have the same complexity.*

It can easily be seen that Proposition 4 does not hold for non-periodic sequences in general. For example the sequence $(aa\cdots ab)$ of length l has complexity $l-1$, its reciprocal $(ba\cdots aa)$ has complexity 1.

Feedback Functions of the Maximum Order FSR Equivalent

The maximum order feedback shift register equivalent of a sequence $\underline{s}$ is defined as the FSR of length $c(\underline{s})$ and a feedback function such that the FSR can generate the sequence $\underline{s}$. We now restrict ourselves to sequences of characters which are elements from some finite field $GF(q)$. For finite field sequences it is customary to use the truth table to derive an analytical expression for the feedback function. In general the truth table of a sequence will not be specified for all q^c possible entries, if c is the maximum order complexity of the sequence. This is due to the fact that not necessarily all q^c possible FSR states occur in a particular sequence. The consequence is that there exists an entire class $\Phi_{\underline{s}}$ of feedback functions which all give rise to the same sequence $\underline{s}$. For this class of feedback functions the following result is obtained:

Proposition 5 *Let $\Phi_{\underline{s}}$ denote the class of feedback functions of the maximum order feedback shift register equivalent of the periodic sequence $\underline{s}$ over $GF(q)$, where $\underline{s}$ has complexity c and period p. The number $|\Phi_{\underline{s}}|$ of functions in the class $\Phi_{\underline{s}}$ satisfies:*

$$|\Phi_{\underline{s}}| = q^{q^c - p}.$$

So, unlike with linear complexity where the feedback function is unique for periodic sequences, $\Phi_{\underline{s}}$ in general contains more than one function and one is able to search for functions exhibiting certain properties such as non-singularity, the least order product function or the function with the least number of terms.

2.2 The Maximum Order Complexity Profile

In [2] Blumer et al. describe a linear-time and -memory algorithm to build a *Directed Acyclic Word Graph* (DAWG) from a given string of letters, using a mechanism of suffixpointers as described in [13]. This DAWG is then used to recognize all substrings (or words) in the string.

The DAWG consists of at most $2l$ nodes connected by at most $3l$ edges, where l is the length of the string. The nodes represent equivalence classes of substrings and the edges are labeled with string letters. An edge points from one node to another if and only if the first equivalence class contains a substring, which extended with the edge's letter belongs to the other equivalence class. The suffix pointer is an edge which points from a node to the node representing the equivalence class with the longest common suffix of all strings of the first node's equivalence class. Two substrings are defined to be equivalent if and only if their endpoint sets are equal. An endpoint set of a given substring is defined as the set containing all positions within a string where the given substring ends. The edges of a DAWG are divided into primary and secondary edges. An edge is called primary if and only if it belongs to the primary path, which is the longest path from the source to a node. With the length of a path the number of edges in that path is meant. The depth $d(\vartheta)$ of a node ϑ is the length of the primary path from the source to that node. The

maximum depth $\hat{d}(\underline{s})$ of a sequence $\underline{s}$ is defined as the maximum of the depths of all the nodes with more than one outgoing edges (denoted by the set of branchnodes $BN(\underline{s})$) of the DAWG of $\underline{s}$.

The following proposition relates the complexity of a sequence to its maximum depth in the DAWG.

Proposition 6 *The complexity $c(\underline{s})$ of a sequence $\underline{s}$ with characters from some finite alphabet $\mathcal{A}$ satisfies:*

$$c(\underline{s}) = \begin{cases} 0; & BN(\underline{s}) = \emptyset, \\ \hat{d}(\underline{s}) + 1; & else. \end{cases}$$

It appears that the DAWG is an efficient tool to determine the maximum order complexity profile of a given sequence.

Proposition 7 *Blumer's algorithm can be used to determine the complexity profile of a sequence $\underline{s}$ with characters from some finite alphabet $\mathcal{A}$ in linear time and memory.*

As Blumer et al. did in their paper, it should be noted that the linearity of their algorithm is with regard to the total processing time related to the length of the sequence.

Blumer's algorithm can be used for a variety of other purposes by postprocessing the DAWG. Examples are: determing the period of a periodic sequence, finding a subsequence in a given sequence (linear in the subsequence length), generating a given sequence based on the least number of observed characters, etc.

The algorithm introduced by Karlin et al. [7] cannot be compared with Blumer's algorithm as it seems to be a two-pass algorithm. This fact renders Karlin's algorithm as unsuitable for the purpose of determining the complexity profile of a given sequence.

2.3 The Typical Complexity Profile

In this section the behaviour of the maximum order complexity profile is viewed at. Let $\underline{s} = (\alpha_0, \alpha_1, \ldots, \alpha_{l-1})$ be a sequence of length l and complexity $c(\underline{s})$ with characters $\alpha_i \in \mathcal{A}$, where the character alphabet $\mathcal{A}$ is some finite set. In the sequel we will use c_l to denote the complexity $c(\alpha_0, \alpha_1, \ldots, \alpha_{l-1})$.

As with linear complexity, the value $l/2$ forms a boundary value, which determines whether the complexity profile jumps to a higher value or remains the same.

Proposition 8 *If the sequence $\underline{s}$ of length l, mentioned above, has complexity c_l, then the value of the complexity c_{l+1} of the sequence, extended with α_l, is given by:*

$$\begin{aligned} c_{l+1} &= c_l, && c_l \geq l/2 \\ &\leq l - c_l, && c_l < l/2 \\ &\geq c_l + 1, && 2c_l < l < c_l + a^{c_l} \\ &\geq l + 1 - a^{c_l}, && l \geq c_l + a^{c_l}, \end{aligned}$$

where $a = |\mathcal{A}|$ is the cardinality of the character alphabet.

Proposition 8 shows exactly how the complexity profile jumps from one value to some other value if a sequence is extended with some character α_l, a phenomenon illustrated by Figure 1. Another way to look at the jump behaviour of the com-

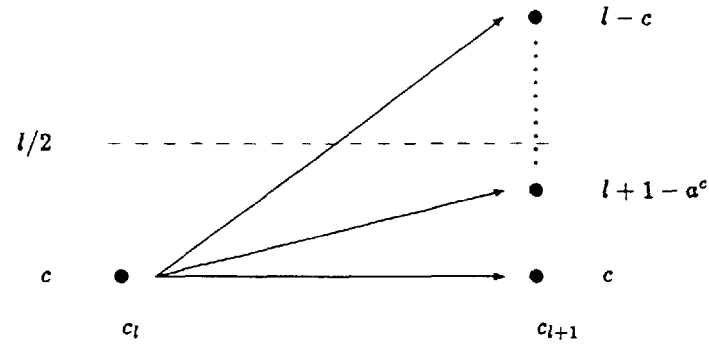

Figure 1: Jumps in the complexity profile.

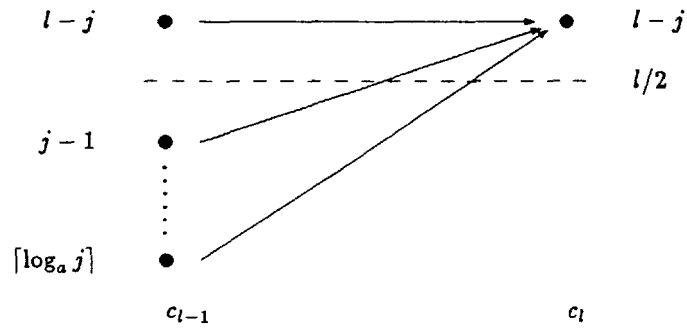

Figure 2: Backwards relationship of complexity values.

plexity profile is the following. Assume that $c_l = l - j$ and $c_{l-1} = j - n$, for positive n. What are the possible values of j and l for certain values of n, i.e. what are the possible values of c_{l-1}? The answer to this question is given by the following proposition.

Proposition 9 *Let $c_l = l - j$ and $c_{l-1} = j - n$, with $1 \leq n \leq j < l$ then n is additionally restricted by the inequality $n \leq j - \log_a j$ and hence $c_{l-1} \geq \log_a j$*

Figure 2 illustrates this backwards relationship between successive complexity values.

2.4 Sequence Analysis and FSR Synthesis

Using various statistical models to describe the behaviour of maximum order complexity with increasing sequence length, in [6] it is demonstrated that the expected maximum order complexity $\bar{c}_l$ of a random sequence of length l with characters from a finite alphabet $\mathcal{A}$ with cardinality a is given by

$$\mathcal{E}(c_l) \approx 2\log_a l.$$

In fact this turns out to be an upperbound, whereas the lowerbound is proven to be $\log_a l$. This result itself was already known for some time, see e.g. [7] and [1]. Consequently, the maximum order complexity profile of random sequences follows the $2\log_a l$ curve. This fact in itself can be used to judge the randomness of given sequences. For example, DeBruijn sequences of order n (see e.g. [4]) have a maximum order complexity of n and are therefore clearly qualified as non-random.

In order to construct the shortest feedback shift register which generates a given sequence of length l one first determines its maximum order complexity and then one determines its feedback function. The first operation has order proportional to l and is expected to yield a complexity value of $2\log l$. The second operation, which can be performed with standard techniques (see [6]), has order $c2^c$ for the binary case. Hence, the expected order of the FSR synthesis procedure is $2l^2\log l$ (as opposed to $l\log l$ for DeBruijn sequences). This expected order clearly limits the feasibility of general FSR resynthesis to moderate length sequences.

3 Conclusions

Our research has highlighted the problem of finding the shortest feedback shift register which generates a given sequence with characters from some finite alphabet. We have focussed here on the absolutely shortest FSR, regardless of its feedback function, which could be highly nonlinear. To this end, a new complexity measure has been introduced, called the maximum order complexity, as opposed to the first order, or linear complexity, the second order, or quadratic complexity, etc. The basic properties of maximum order complexity have been shown and, in fact, it has been demonstrated that the maximum order complexity is strongly connected with nonlinear feedback shift registers. We believe that our results provide a new contribution to the theory of nonlinear feedback shift registers and a better understanding of their functioning.

The practical import of maximum order complexity has been enhanced by the identification of an efficient algorithm for obtaining the maximum order complexity profile of arbitrary sequences. It has been shown that the maximum order complexity profile can be determined in linear time and memory, using Blumer's algorithm.

By considering the complexity of random sequences, the typical behaviour of the maximum order complexity profile has been clarified.

The consequences for the analysis of pseudo-random sequences and the resynthesis with feedback shift registers have been shown, viz. the total effort to determine the shortest FSR equivalent is of the order $2l^2 \log_2 l$ for the binary case.

Concluding, we can say that the maximum order complexity profile is a useful new tool for judging the randomness of sequences. For example, it declares DeBruijn sequences as non-random, whereas these sequences are considered highly complex according to Lempel and Ziv and some of these sequences are also considered complex according to Rueppel's linear complexity profile.

References

[1] R. Arratia, L. Gordon and M. S. Waterman. "An Extreme Value Theory for Sequence Matching", *The Annals of Statistics*, vol. 14, no. 3, pp. 971–993, 1986.

[2] A. Blumer, J. Blumer, A. Ehrenfeucht, D. Haussler and R. McConnell. "Linear Size Finite Automata for the Set of all Subwords of a Word: An Outline of Results", *Bul. Eur. Assoc. Theor. Comp. Sci.*, no. 21, pp. 12–20, 1983.

[3] D. W. Davies and W. L. Price. *Security for Computer Networks*, John Wiley & Sons, Inc., Chichester, 1984.

[4] H. Fredricksen. "A survey of full-length nonlinear shift register cycle algorithms", *SIAM Rev.*, vol. 24, pp. 195–221, April 1982.

[5] S. W. Golomb. *Shift Register Sequences*, Holden–Day Inc., San Francisco, 1967.

[6] C. J. A. Jansen. *Investigations On Nonlinear Streamcipher Systems: Construction and Evaluation Methods*, PhD. Thesis, Technical University of Delft, Delft, 1989.

[7] S. Karlin, G. Ghandour, F. Ost, S. Tavare and L. J. Korn. "New Approaches for Computer Analysis of Nucleic Acid Sequences", *Proc. Natl. Acad. Sci. USA*, vol. 80, pp. 5660–5664, September 1983.

[8] A. G. Konheim. *Cryptography*, John Wiley & Sons, Inc., New York, 1981.

[9] A. Lempel and J. Ziv. "On the Complexity of Finite Sequences", *IEEE Trans. on Info. Theory*, vol. IT–22, no. 1, pp. 75–81, January 1976.

[10] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error Correcting Codes*, Amsterdam, North-Holland, 1978.

[11] J. L. Massey. "Shift-Register Synthesis and BCH Decoding", *IEEE Trans. on Info. Theory*, vol. IT–15, January 1969.

[12] C. H. Meyer and S. M. Matyas. *Cryptography: A New Dimension in Computer Data Security*, John Wiley & Sons, New York, 1982.

[13] E. M. McCreight. "A space-economical suffix tree construction algorithm", *JACM*, vol. 23, no. 2, pp. 262–272, April 1976.

[14] R. A. Rueppel. *New Approaches to Stream Ciphers*, PhD. Thesis, Swiss Federal Institute of Technology, Zurich, 1984.

[15] C. E. Shannon. "Communication Theory of Secrecy Systems", *Bell Systems Technical Journal*, vol. 28, pp. 656–715, October 1949.