The Self-Shrinking Generator *

Willi Meier[1] and Othmar Staffelbach[2]

[1] HTL Brugg-Windisch
CH-5200 Windisch, Switzerland
meierw@htlulx.htl-bw.ch

[2] Gretag Data Systems AG
CH-8105 Regensdorf, Switzerland

Abstract. A construction of a pseudo random generator based on a single linear feedback shift register is investigated. The construction is related to the so-called shrinking generator and is attractive by its conceptual simplicity. The lower bounds that are provided for period, linear complexity and known cryptanalytic attacks allow for efficient practical implementations at a reasonable scale.

1 Introduction

In [1] a new pseudo random sequence generator, the so-called *shrinking generator*, has been suggested by Coppersmith, Krawczyk and Mansour for potential use in stream cipher applications. The shrinking generator is attractive by its conceptual simplicity as it combines only two LFSRs in a simple way. One is tempted to conjecture that such a simple construction might be insecure. However no successful cryptanalytic attack has been publicly reported so far.

In this paper we present an apparently simpler structure using only one LFSR whose output sequence is shrunken in a similar way as is done for the shrinking generator. As the shrinking of the LFSR-sequence is achieved under the control of the LFSR itself, the generator will be called *self-shrinking generator*.

Recall that the shrinking generator [1] uses two binary LFSRs, say LFSR 1 and LFSR 2, as basic components. The pseudo random bits are produced by shrinking the output sequence of LFSR 1 under the control of LFSR 2 as follows: The output bit of LFSR 1 is taken if the current output of LFSR 2 is 1, otherwise it is discarded. For the self-shrinking generator we suggest to use only one LFSR. Instead of output bits, pairs of output bits are considered. If a pair happens to take the value 10 or 11, this pair is taken to produce the pseudo random bit 0 or 1, depending on the second bit of the pair. On the other hand if a pair happens to be 01 or 00, it will be discarded. The key consists of the initial state of the LFSR and preferably also of the LFSR feedback logic. For practical applications it is assumed that the feedback connection is to produce maximal length LFSR-sequences.

* A version of this paper will appear in the proceedings of a symposium in honor of James L. Massey on the occasion of his 60th birthday.

The self-shrinking mechanism of an LFSR might be compared with the self-decimation of an LFSR as introduced in [4]. As mentioned in [4], the self-decimated sequence cannot be directly applied for stream enciphering. As the decimation intervals are revealed by the output sequence, one can derive the original LFSR-sequence at fixed positions from the self-decimated sequence. Thus the original LFSR-sequence can be computed by solving linear equations. For a shrunken or self-shrunken sequence one also sees certain output bits of the original LFSR-sequence, but one does not know the size of the gaps between the known bits.

It turns out that the self-shrinking generator and the shrinking generator are closely related to each other. In fact, it will be shown in Section 2 that the self-shrinking generator can be implemented as a shrinking generator, and conversely, that the shrinking generator can be implemented as a self-shrinking generator. The latter implementation however cannot be accomplished with a maximum length LFSR. Thus the self-shrinking generator has its main interest in implementing the shrinking principle at lower hardware costs. According to [1], the effective key size of the shrinking generator, measured in terms of the complexity of known cryptanalytic attacks, is roughly half of the maximum possible key size. In view of the presently known cryptanalytic attacks (see Section 5) the effective key size of the self-shrinking generator can be estimated to be more than 80% of the maximum possible value.

It is difficult to give a general and reliable measure for the cryptographic quality of pseudo random sequences being applied in stream cipher systems. Certainly well known are the classical measures, period and linear complexity. For a secure design one should have proofs or at least strong practical evidence that these parameters are large enough to withstand the more generic attacks like the Berlekamp-Massey LFSR synthesis algorithm [3]. For a self-shrinking generator implemented with a maximum length LFSR of length N, it is proved in Section 3 that the period and the linear complexity are lower bounded by $2^{\lfloor N/2 \rfloor}$ and $2^{\lfloor N/2 \rfloor - 1}$, respectively. Furthermore in Section 4 strong evidence is provided that the period is in fact 2^{N-1} for $N > 3$, and that the linear complexity is very close to that value. Therefore it is easy to implement the self-shrinking generator to satisfy sufficiently large proved lower bounds for period and linear complexity.

The experimental results in Section 4 reveal another interesting fact, namely that the linear complexity does not exceed the value $2^{N-1} - N + 2$. This can be considered as an algebraic property of the shrunken LFSR-sequence. The original LFSR-sequence has a rich algebraic structure. For being applicable as pseudo randomizer for cryptographic purposes it is necessary to destroy most of the algebraic structure—in particular the property of satisfying a short linear recursion.

For the self-shrinking generator, the fact that it is unknown at which positions the LFSR-sequence is shrunken and that the shrinking is controlled by the LFSR itself suggest that most of the algebraic structure of the original LFSR-sequence has been destroyed. Thus the above mentioned upper bound on the linear com-

plexity appears to be surprising. Proving this fact as well as the conjecture that 2^{N-1} is the minimum period remain as open problems. These problems deal with elementary manipulations on LFSR-sequences, for which a thorough mathematical theory is available.

2 Shrinking and Self-Shrinking

Self-shrinking can be applied to arbitrary binary sequences. The original sequence $\mathbf{a} = (a_0, a_1, a_2, \ldots)$ is considered as a sequence of pairs of bits $((a_0, a_1), (a_2, a_3), \ldots)$. If a pair (a_{2i}, a_{2i+1}) equals the value $(1, 0)$ or $(1, 1)$, it is taken to produce the pseudo random bit 0 or 1, respectively. On the other hand, if the pair is equal to $(0, 0)$ or $(0, 1)$, it will be discarded, which means that it will not contribute an output bit to the new sequence $\mathbf{s} = (s_0, s_1, s_2, \ldots)$.

Self-shrinking is in particular intended to be applied to pseudo random sequences in order to produce new pseudo random sequences of (potentially) better cryptographic quality. We especially analyze the situation where the original sequence $\mathbf{a}$ is generated by an LFSR. For a cryptographic application the key consists of the initial state of the LFSR. Preferably the feedback connection is variable and also part of the key. The self-shrunken sequence $\mathbf{s}$ can be considered as being obtained from the original sequence $\mathbf{a}$ by discarding certain bits. In the average 3/4 of the bits are expected to be omitted. Hence the data rate of the original sequence is reduced by the factor 4.

It appears to be natural to ask the question whether the self-shrinking generator can be implemented as a special case of the shrinking generator. To show that this is in fact the case, let $\mathbf{a} = (a_0, a_1, a_2, \ldots)$ be the sequence produced by an LFSR of length N defining a self-shrinking generator. According to the self-shrinking rule, the sequence $(a_0, a_2, a_4, \ldots)$ effects the output control, and $(a_1, a_3, a_5, \ldots)$ defines the sequence being controlled. Both sequences can be produced by the original LFSR when loaded with the initial states $(a_0, a_2, \ldots, a_{2N-2})$, or $(a_1, a_3, \ldots, a_{2N-1})$ respectively. This implies that the self-shrinking generator can be implemented as a shrinking generator with two LFSRs having identical feedback connections.

Conversely, we will show that the shrinking generator can be implemented as a special case of the self-shrinking generator. To this end, consider an arbitrary shrinking generator defined by two linear shift registers LFSR 1 and LFSR 2 with feedback polynomials $f(x)$, and $g(x)$, respectively. Furthermore, let $\mathbf{b} = (b_0, b_1, b_2, \ldots)$ and $\mathbf{c} = (c_0, c_1, c_2, \ldots)$ denote the corresponding LFSR output sequences. Then, by applying the self-shrinking rule to the interleaved sequence $\mathbf{a} = (c_0, b_0, c_1, b_1, \ldots)$, the original output sequence of the shrinking generator is reproduced. On the other hand, it can be shown that the sequence $\mathbf{a}$ can be produced by an LFSR with feedback polynomial $f(x^2)g(x^2) = f(x)^2 g(x)^2$. This implies that the shrinking generator has an equivalent implementation as a self-shrinking generator.

The investigations on the shrinking generator in [1] assume that the two LFSRs involved are independent, e.g., that their periods are coprime. Therefore

the results in [1] on period and linear complexity do not apply to the self-shrinking generator. For obtaining corresponding results for the self-shrinking generator, a different approach will be required.

3 Period and Linear Complexity of Self-Shrunken Maximum Length LFSR-sequences

We now establish lower and upper bounds on period and linear complexity of self-shrunken sequences generated by maximum length LFSRs (m-LFSRs).

3.1 Period

Let $\mathbf{a} = (a_0, a_1, a_2, \ldots)$ be the output sequence of a non-trivially initialized m-LFSR of length N. Hence $\mathbf{a}$ is a sequence with period $2^N - 1$. The self-shrunken sequence will also be periodic. In fact, after $2(2^N - 1)$ bits of the original sequence, the sequence of pairs $(a_0, a_1), (a_2, a_3), \ldots, (a_{2^N-2}, a_0), (a_1, a_2), \ldots, (a_{2^N-3}, a_{2^N-2})$ has been processed, and the next pair will be (a_0, a_1) again. Hence the shrunken sequence is repeating. Within this period each possible output pair $(a_i, a_{i+1}), 0 \leq i < 2^N - 1$, of the original LFSR-sequence has occurred exactly once. As is well-known, within the period of a m-LFSR-sequence each of the pairs 01, 10, and 11 appears exactly 2^{N-2} times, and the pair 00 appears $2^{N-2} - 1$ times. By the definition of the shrinking rule, it follows that 2^{N-1} is a period of the shrunken sequence. Moreover, as the pairs 10 and 11 occur equally often, the shrunken sequence must be balanced. As the shrunken sequence is repeating after 2^{N-1} bits, it must be purely periodic with period $p = 2^{N-1}$, i.e., $s_n = s_{n+p}$ for all $n > 0$. This implies that the smallest period P of $\mathbf{s}$ must divide 2^{N-1}. Summarizing we obtain

Proposition 1. *Let* $\mathbf{a}$ *be an* m*-LFSR-sequence generated by an LFSR of length* N *and let* $\mathbf{s}$ *be the self-shrunken sequence obtained from* $\mathbf{a}$*. Then* $\mathbf{s}$ *is a balanced sequence whose period divides* 2^{N-1}.

A lower bound on the period of a shrunken m-LFSR-sequence is given in the following theorem.

Theorem 2. *The period* P *of a self-shrunken maximum length LFSR-sequence produced by an LFSR of length* N *satisfies*

$$P \geq 2^{\lfloor N/2 \rfloor}. \tag{1}$$

Proof. Let us first consider the case when N is even, and let $n = N/2$. Since the feedback connection of the LFSR is chosen to produce maximum length sequences, every nonzero N-bit word appears exactly once when scanning the LFSR-sequence with a window of length N over the full period. In view of the self-shrinking, we consider the sequence $\mathbf{a}$ being scanned over the double period with increments by two bits. As the period is odd, the same N-bit patterns

occur (possibly in different order) as if the sequence were scanned over one period with one bit increments. By the maximum length property, the N-bit pattern $(1, x_1, 1, x_2, \ldots, 1, x_n)$ appears in the original sequence for every choice of $(x_1, x_2, \ldots, x_n)$. It follows that every n-bit pattern appears in the shrunken sequence when scanning it with window size n.

If a sequence of period P is scanned over an interval of arbitrary length, at most P different patterns can occur (independent of the window size). As the shrunken sequence contains all 2^n patterns of length n, it follows that the inequality $P \geq 2^n$ must hold. This proves the theorem for the case when N is even. For odd N let $n = (N-1)/2$. Then the $(N-1)$-bit pattern $(1, x_1, 1, x_2, \ldots, 1, x_n)$ appears (twice) when scanning the original sequence. The rest of the proof is similar as in the case when N is even. □

3.2 Linear Complexity

For purely periodic sequences the linear complexity L is equal to the degree of the minimal polynomial $f(x)$. Recall that $f(x)$ is defined as the characteristic polynomial of the shortest linear recursion satisfied by the sequence (see [2]). Furthermore, the minimum period of the sequence is the smallest positive integer P such that $f(x)$ divides $x^P - 1$. For a self-shrunken m-LFSR-sequence the linear complexity satisfies a lower bound as given in Theorem 3.

Theorem 3. *The linear complexity L of a self-shrunken maximum length LFSR-sequence produced by an LFSR of length N satisfies*

$$L > 2^{\lfloor N/2 \rfloor - 1}. \tag{2}$$

Proof. By Proposition 1 and Theorem 2 the period P of a self-shrunken m-LFSR-sequence s divides 2^{N-1}, i.e., is of the form $P = 2^a$ for some integer $a \geq \lfloor N/2 \rfloor$. Hence over $GF(2)$, $x^P - 1$ can be written as $x^P - 1 = (x-1)^{2^a}$. Thus the condition $f(x) \mid (x^P - 1)$ implies that $f(x)$ is of the form $f(x) = (x-1)^L$ where L is the linear complexity of the sequence s. We claim that $L > 2^{a-1}$.

Suppose to the contrary that $L \leq 2^{a-1}$. Then $f(x) = (x-1)^L$ would divide $(x-1)^{2^{a-1}} = x^{2^{a-1}} - 1$. Thus $x^{2^{a-1}} - 1$ would be the characteristic polynomial of a recursion satisfied by s. This recursion would be $s_n = s_{n-2^{a-1}}$ which contradicts to the fact that the minimum period is 2^a. □

It is a common assumption in the analysis of the shrinking generator [1] or clock-controlled generators that the two LFSRs involved are independent. This allows for example to decimate the process of generating the output sequence with the period of the controlling LFSR. The output sequence obtained in this way can be considered as a decimated sequence of the controlled LFSR. This allows to apply the theory of LFSR-sequences to derive results on the period and linear complexity of the generated output sequence. This approach cannot be applied to the self-shrinking generator as the controlling and the controlled part cannot be separated from one another. For this reason the exact computation of the period and the linear complexity of a self-shrunken m-LFSR-sequence appears to be difficult. The bounds given in Theorems 2 and 3 are rough estimates.

Experimental results as given in Section 4 support the conjecture that the period P is maximal for LFSR-length $N > 3$, i.e., $P = 2^{N-1}$. For the linear complexity L this would imply that L is bounded by $2^{N-2} < L \leq 2^{N-1}$. Nevertheless the bounds as given in Theorems 2 and 3 are far sufficient for practical applications. For example for $N = 200$, period and linear complexity are proved to be at least 10^{30}.

4 Examples and Experimental Results

By the analysis in Section 3 the period of a self-shrunken m-LFSR-sequence generated by an LFSR of length N is at most 2^{N-1}. So far we have found only one example where the period does not reach this maximum value. This is the m-LFSR of length $N = 3$ defined by the recursion $a_n = a_{n-2} + a_{n-3}$. The corresponding self-shrunken sequence has period only 2 instead of the maximum possible value 4.

Experiments have shown that for all other m-LFSRs of length $N < 20$ the self-shrunken sequences attain maximum period 2^{N-1}. This has been confirmed by exhausting all m-LFSRs of length $N < 20$. Table 1 shows the minimum and the maximum value of the linear complexity taken over all self-shrunken m-LFSRs of given LFSR-length N for $N \leq 15$.

LFSR-length N	*# of m-LFSR*	*Minimum LC*	*Maximum LC*	δ
2	1	2	2	0
3	2	2	3	1
4	2	5	5	3
5	6	10	13	3
6	6	25	28	4
7	18	54	59	5
8	16	118	122	6
9	48	243	249	7
10	60	498	504	8
11	176	1009	1015	9
12	144	2031	2038	10
13	630	4072	4085	11
14	756	8170	8180	12
15	1800	16362	16371	13

Table 1. Minimum and maximum linear complexity of self-shrunken m-LFSRs

Commenting Table 1, we first note that for a sequence with an even number of 1's within the period P, the maximum possible linear complexity is $P - 1$, as $\sum_{i=0}^{P-1} s_{n-i} = 0$. For self-shrunken m-LFSR-sequences, maximum and minimum value of the linear complexity appear to be close to each other and very close to the maximum possible value $2^{N-1} - 1$.

Furthermore Table 1 shows a remarkable property: Except for $N = 4$, the upper bound attained for the linear complexity is $2^{N-1} - \delta$, where $\delta = N - 2$. This upper bound also holds for the exceptional case $N = 4$. Hence, for $2 \leq N \leq 15$, $(x^{2^{N-1}} - 1)/(x - 1)^{N-2}$ is a characteristic polynomial of any self-shrunken m-LFSR-sequence produced by an LFSR of length N. This fact can be viewed as an algebraic property of the self-shrunken LFSR-sequence that persists although most of the algebraic structure of the original m-LFSR-sequence has been destroyed.

5 Cryptanalysis

In this section we discuss some approaches for possible cryptanalytic attacks and their complexities. We start with a general method for reconstructing the original sequence from a known portion of the self-shrunken sequence. This method is not restricted to the case where the original sequence is produced by an LFSR.

Assume that $(s_0, s_1, \ldots)$ is the known portion of the self-shrunken sequence. The bit s_0 is produced by a bit pair (a_j, a_{j+1}) of the original sequence where the index j is unknown. Our aim is to reconstruct the original sequence in forward direction beginning with position j. As we know s_0 we conclude that $a_j = 1$ and $a_{j+1} = s_0$. For the next bit pair (a_{j+2}, a_{j+3}) there remain three possibilities, namely $a_{j+2} = 1, a_{j+3} = s_1$ if the bit pair was used to produce s_1, or the two alternatives $a_{j+2} = 0, a_{j+3} = 0$ and $a_{j+2} = 0, a_{j+3} = 1$ if the bit pair was discarded. For each of the three possibilities there are again three alternatives for the next bit pair. Therefore, for reconstructing n bit pairs, i.e., $N = 2n$ bits, we obtain a total of

$$S = 3^{n-1} \approx 3^{N/2} = 2^{((\log_2 3)/2)N} = 2^{0.79 \cdot N} \tag{3}$$

possible solutions. However the solutions have different probabilities. We explain this fact by considering the above bit pair (a_{j+2}, a_{j+3}). Assuming that the original sequence is purely random, $a_{j+2} = 1$ with probability 1/2. Hence the first alternative has probability 1/2 and the other two cases have probability 1/4. In terms of information theory the uncertainty about the bit pair is

$$H = -(1/2)\log_2(1/2) - (1/4)\log_2(1/4) - (1/4)\log_2(1/4) = 3/2.$$

As for the reconstruction the individual bit pairs are supposed to be independent from each other, the total entropy for n bit pairs is $3n/2$. Therefore the optimum strategy for reconstructing N bits of the original sequence has average complexity $2^{3N/4}$. For example, for $N = 200$, this complexity is equivalent to an exhaustive search over a key of size 150 bit.

So far we did not take into account that the original sequence is produced by an LFSR. For cryptographic applications the key consists of the initial state and preferably also of the LFSR feedback connection. In order to assess the security we assume that the feedback connection is known. With this assumption we estimate the difficulty of finding the initial state (or the key) of the LFSR. For

the above method of finding the key the average complexity is upper bounded by $2^{3N/4}$, where N is the length of the LFSR. If there are only few feedback taps or if they are concentrated around few locations, there are cases where faster attacks are possible, as will be shown below. On the other hand, if we exclude such special situations we know of no better method than reconstructing the initial state of the LFSR as described above.

Suppose for example that the LFSR only has two feedback taps (which is the smallest number of feedback taps for a m-LFSR). Then the feedback relation can be written as $a_k + a_{k+t} + a_{k+t+s} = 0$, for all $k \in \mathbf{N}$. Let a_j be the bit of the original sequence which determines the first known bit, say s_0, of the shrunken sequence. Our aim is to do an exhaustive search over the two m-bit blocks

$$\mathcal{B}_1 = (a_j, a_{j+1}, \ldots, a_{j+m-1})$$
$$\mathcal{B}_2 = (a_{j+t}, a_{j+t+1}, \ldots, a_{j+t+m-1})$$

of suitably chosen size m. For every choice of the two blocks the third block

$$\mathcal{B}_3 = (a_{j+t+s}, a_{j+t+s+1}, \ldots, a_{j+t+s+m-1})$$

is determined by the feedback relation. By self-shrinking there result three bit strings. The known segment of the self-shrunken sequence is scanned for the occurrence of these strings. For the correct choice of the m-bit blocks the three strings are expected to be about $s/4$ or $t/4$ bits apart from each other.

We call a block pair a solution if the three strings can be found at suitable positions. We investigate the problem regarding the number of solutions that are to be expected. According to (3) there are about $3^{m/2}$ solutions for $\mathcal{B}_1$. If one knows the position of the substring in the shrunken sequence which is produced by the second block $\mathcal{B}_2$, one again has about $3^{m/2}$ solutions for $\mathcal{B}_2$. As this position is not exactly known, the number of solutions for $\mathcal{B}_2$ is slightly larger. Thus we conclude that there are at least about $3^{m/2} \cdot 3^{m/2} = 3^m$ solutions for the pair $(\mathcal{B}_1, \mathcal{B}_2)$. By the same argument we conclude that there are at least about $3^{m/2}$ solutions for $\mathcal{B}_3$. It follows that with probability about $p = 3^{m/2}/2^m$, a random block $\mathcal{B}_3$ is compatible with the shrunken sequence. Thus the number of solutions for the pair $(\mathcal{B}_1, \mathcal{B}_2)$ is reduced by the factor p due to the recurrence relation. Therefore there remain about

$$T = 3^m \frac{3^{m/2}}{2^m} = \frac{3^{3m/2}}{2^m} = 2^{[3(\log_2 3)/2-1]m} = 2^{1.38 \cdot m} \tag{4}$$

solutions. For finding these solutions a search over 2^{2m} block pairs is necessary.

In a similar way, with complexity 2^{2m}, we do an exhaustive search over m-bit blocks

$$\mathcal{B}'_1 = (a_{j-1}, a_{j-2}, \ldots, a_{j-m})$$
$$\mathcal{B}'_2 = (a_{j+t-1}, a_{j+t-2}, \ldots, a_{j+t-m})$$

in reverse direction from position j, or $j+t$, respectively. As for $(\mathcal{B}_1, \mathcal{B}_2)$ there remain $T = 2^{1.38 \cdot m}$ solutions for $(\mathcal{B}'_1, \mathcal{B}'_2)$. Every solution for $(\mathcal{B}_1, \mathcal{B}_2)$ and $(\mathcal{B}'_1, \mathcal{B}'_2)$

defines $4m$ bits of the LFSR-sequence. Since N bits are required for reconstructing the original LFSR-sequence, we choose $m = N/4$. Thus the complexity of the search is $2 \cdot 2^{N/2}$ with a possibility of $T^2 = 2^{[3(\log_2 3)/2-1]N/2} = 2^{0.69 \cdot N}$ remaining solutions. The correct solution is singled out by trying all these possible solutions. This second exhaustive search obviously has complexity $2^{0.69 \cdot N}$ which dominates the over all complexity of the attack. Thus the fact that the LFSR has only two feedback taps allows an attack which is slightly faster than the general method whose complexity is $2^{0.75 \cdot N}$.

The described method is a divide and conquer attack. The key is divided into two block pairs $(\mathcal{B}_1, \mathcal{B}_2)$ and $(\mathcal{B}'_1, \mathcal{B}'_2)$, and the search for each block pair is done individually. It seems straightforward to extend the attack by searching for k rather than for two different m-bit block pairs. The complexity then would be $k2^{2m}$ with $(2^{1.38 \cdot m})^k$ possible solutions remaining. Each solution would determine $2km$ bits of the LFSR-sequence. In order to obtain N bits we would choose $k = N/(2m)$. For $k > 2$ the initial search has lower complexity. However the over all complexity is still dominated by the number of solutions which is $(2^{1.38 \cdot m})^{N/(2m)} = 2^{0.69 \cdot N}$ as for $k = 2$.

It turns out that the attack is less effective if the number f of feedback taps increases. Corresponding to the feedback tap positions at the LFSR we would search for tuples $(\mathcal{B}_1, \ldots, \mathcal{B}_f)$ of m-bit blocks. Instead of (4), a number

$$T = (3^{m/2})^f \frac{3^{m/2}}{2^m} = 3^{((f+1)/2)m} 2^{-m} = 2^{[(\log_2 3)(f+1)/2-1]m} \tag{5}$$

of candidate solutions would remain after the search. Following the idea of divide and conquer we would search for at least $k = 2$ such tuples. For $k = 2$ these would determine $2fm$ bits of the original LFSR-sequence. This suggests to choose $m = N/(2f)$. Thus the complexity of the search is again $2 \cdot 2^{N/2}$ but with a possibility of

$$T^2 = 2^{[(\log_2 3)(f+1)/2-1]N/f} = 2^{[(\log_2 3)(1/2-1/(2f)]N} \tag{6}$$

solutions. For $f = 4$ this quantity is $2^{0.74 \cdot N}$, and the asymptotic value, as f increases, is $2^{((\log_2 3)/2)N} = 2^{0.79 \cdot N}$. This coincides with the number of solutions (3) obtained for the general method.

The feasibility of the attack is further limited as the blocks become shorter. For shorter blocks the corresponding shrunken strings are more likely to appear accidentally in the shrunken sequence. This has the effect that it is more difficult to link the blocks with the corresponding positions in the shrunken sequence. Hence more incorrect solutions are likely to be accepted in the initial search.

The above cryptanalytic investigations give no means to break the self-shrinking generator, if we exclude special situations. Our best method for reconstructing the initial state of an LFSR of length N has complexity $2^{0.75 \cdot N}$, even if the feedback logic is known. If the feedback connection is also part of the key, the reconstruction of the initial state has to be combined with an exhaustive search over all (primitive) feedback connections. Therefore the complexity of the attack is increased by the factor $\varphi(2^N - 1)/N$, which for large N may be

approximated by 2^N. Hence the total complexity of the attack is approximately $2^{1.75 \cdot N}$. As the key size is about $2N$, the effective key size is more than 80% of the maximum possible value.

Acknowledgement

We are grateful to Christoph Günther and one of the referees for helpful comments.

References

1. D. Coppersmith, H. Krawczyk, Y. Mansour, *The Shrinking Generator*, Crypto'93, to appear.
2. S.W. Golomb, *Shift Register Sequences*, Aegean Park Press, 1982.
3. J.L. Massey, *Shift Register Synthesis and BCH Decoding*, IEEE Transactions on Information Theory, Vol. IT-15, pp. 122–127, 1969.
4. R.A. Rueppel, *When Shift Registers Clock Themselves*, Advances in Cryptology—Eurocrypt'87, Proceedings, pp. 53–64, Springer-Verlag, 1988.