J. Cryptology (2003) 16: 41–68
DOI: 10.1007/s00145-002-9925-1

Journal of
CRYPTOLOGY

Edit Probability Correlation Attacks on Stop/Go Clocked Keystream Generators*

Jovan Dj. Golić and Renato Menicocci
Rome CryptoDesign Center, Gemplus,
Via Pio Emanuelli 1, 00143 Rome, Italy
{jovan.golic,renato.menicocci}@gemplus.com

Communicated by Andrew Odlyzko

Received December 1999 and revised June 2002
Online publication 6 December 2002

Abstract. Divide-and-conquer correlation attacks on the alternating step generator, the bilateral stop/go generator, and the alleged A5 generator are proposed. They are based on appropriately defined edit probabilities incorporating the stop/go clocking in these generators. Recursive algorithms for the efficient computation of the edit probabilities are derived. It is shown how the edit probabilities can be used to mount statistically optimal correlation attacks on the corresponding subsets of stop/go clocked shift registers. By using a statistical hypothesis testing method for estimating the underlying false alarm probability, it is argued that the minimum output sequence length required to be known for a successful attack is linear in the total length of the targeted shift registers. This is illustrated by experimental attacks on the alternating step generator and the bilateral stop/go generator composed of relatively short shift registers.

Key words. Stream ciphers, Stop/go clocked shift registers, Edit probability, Correlation attack

1. Introduction

A common type of keystream generators for additive stream cipher applications consists of a number of irregularly clocked linear feedback shift registers (LFSRs) that are combined by a function, which can even be linear and memoryless. Such generators can produce sequences with long period, high linear complexity, and good statistical properties (e.g., see [10]). On the other hand, they may in principle be vulnerable to certain secret key reconstruction attacks (for a survey, see [12], [13], and [4]). Typically, the

* Parts of this work were presented at the poster session of Eurocrypt '98, Espoo, Finland, June 1998, at Sequences and Their Applications - SETA '98, Singapore, December 1998, and at the 7th IMA Conference on Cryptography and Coding, Cirencester, England, December 1999. This work was done while Jovan Dj. Golić was with the School of Electrical Engineering, University of Belgrade, Yugoslavia, and Renato Menicocci was with Fondazione Ugo Bordoni, Rome, Italy.

attacks require an exhaustive search over the initial states of a subset of the LFSRs and are hence feasible only if the effective secret key controlling these initial states is short.

The stop-and-go (stop/go) clocking is interesting particularly for high speed applications. At any time, a stop/go shift register is clocked once if the clock-control input bit is equal to one (or zero) and is not clocked at all otherwise. The best known examples of keystream generators incorporating stop/go LFSRs are the stop/go cascades [10], the alternating step generator (ASG) [11], the bilateral stop/go generator (BSGG) [14], [15], and the alleged A5 generator (see [13]).

In this paper divide-and-conquer correlation attacks on the ASG, the BSGG, and the alleged A5 are investigated. Their objective is to recover the secret-key-controlled LFSR initial states from a known segment of the keystream sequence in the known plaintext scenario, by targeting a chosen subset of the LFSRs whose initial states are tested exhaustively. The LFSR feedback polynomials are assumed to be known. The correlation attacks are based on appropriate edit probabilities as measures of correlation. Note that edit distance and edit probability correlation attacks on irregularly clocked shift registers, which are clocked at least once at a time, are introduced in [7] and [9], respectively. Regarding the stop/go clocking, a specific edit distance correlation attack on the ASG is proposed in [6]. The main problems considered in this paper are how to define the edit probabilities, how to compute them efficiently, and how to estimate the known keystream sequence length required for a successful correlation attack.

The ASG consists of two stop/go clocked binary LFSRs, LFSR$_1$ and LFSR$_2$, and a regularly clocked clock-control binary LFSR, LFSR$_3$. At each time, the clock-control bit defines which of the two LFSRs is clocked, and the output sequence is obtained as the bitwise sum of the two stop/go clocked LFSR sequences (see Section 2.1). It is shown in [11] that the initial state of LFSR$_3$ can be recovered by a specific divide-and-conquer attack based on the fact that if and only if the guess about the initial state of LFSR$_3$ is correct, then the first (binary) derivative of the output sequence can be de-interleaved into the first (binary) derivatives of regularly clocked LFSR$_1$ and LFSR$_2$ sequences.

The target of the edit distance correlation attack [6] on the ASG are the initial states of LFSR$_1$ and LFSR$_2$ combined. The objective of the edit probability correlation attack considered here are the initial states of each of LFSR$_1$ and LFSR$_2$ individually. The fact that the first derivative of the ASG output sequence is bitwise correlated (with the correlation coefficient $\frac{1}{2}$) to the first derivative of the output sequence of each of the stop/go clocked LFSR$_1$ and LFSR$_2$ indicates that such a divide-and-conquer attack may be possible. Interestingly, it can be shown that the corresponding edit distance correlation attack cannot be successful (see [8]). In principle, this is not surprising as [8], although for a different type of clocking, shows that the edit probability approach may sometimes work when the edit distance one cannot.

The BSGG consists of two binary LFSRs, LFSR$_1$ and LFSR$_2$, which mutually stop/go clock-control each other. More precisely, a clock-control function derives two clock-control bits from the states of the two LFSRs which are used to stop/go clock-control the LFSRs, respectively. The two clock-control bits are never simultaneously equal to zero, so that at each time at least one of the two LFSRs is clocked. The output sequence is formed as the bitwise sum of the two stop/go clocked LFSR sequences (see Section 2.2). No attacks on such a structure are reported in the open literature. Our objective here is to propose an edit probability correlation attack on one of the LFSRs. The fact that the first derivative of the BSGG output sequence is bitwise correlated to the first derivative of the output sequence of each of the stop/go clocked LFSR_1 and LFSR_2 (with the correlation coefficient $\frac{1}{5}$) suggests that a divide-and-conquer attack may be possible.

The alleged A5 generator consists of three binary LFSRs which are mutually clocked in the stop/go manner. Middle taps in each of the LFSRs are used to produce the clock-control bits and the clocking rule is such that at least two LFSRs are clocked at each time. The output sequence is formed as the bitwise sum of the three stop/go clocked LFSR sequences (see Section 2.3). This keystream generator along with a re-initialization scheme is allegedly used under the name A5 for stream cipher encryption in the GSM standard for digital cellular mobile telephones (see [13]). In [3] and [5] a cryptanalytic approach consisting of several methods for LFSR internal states reconstruction, LFSR initial states reconstruction, and secret session key reconstruction is proposed.[1] The methods include the internal state reconstruction consisting of generating and solving a specific set of linear equations, the internal state reconstruction based on a time–memory tradeoff, and the internal state reversion based on the theory of branching processes. The time–memory tradeoff method is further developed in [1].

Our objective here is to propose an edit probability correlation attack on any two of the three LFSRs. Note that a correlation attack on individual LFSRs is not possible, because in the probabilistic model where the regularly clocked LFSR sequences are assumed to be independent and purely random[2] each stop/go clocked LFSR sequence is statistically independent of the output sequence [3], [5]. The fact that the first derivative of the alleged A5 output sequence is bitwise correlated (with the correlation coefficient $\frac{1}{4}$) to the first derivative of the bitwise sum of any two stop/go clocked LFSR sequences indicates that such a divide-and-conquer attack may be possible. Such an attack is not directly applicable to the GSM version of the alleged A5, because of a very short available keystream sequence.

In Section 2 more detailed descriptions of the ASG, the BSGG, and the alleged A5 are provided. For the ASG, the BSGG, and the alleged A5, the corresponding edit probabilities and the recursive algorithms for their efficient computation are presented in Sections 3, 4, and 5, respectively. The general framework of the correlation attack on all three generators, including the underlying statistical hypothesis testing model, is explained in Section 6. The concrete correlation attacks on the ASG, the BSGG, and the alleged A5 along with the corresponding experimental results are presented in Sections 7, 8, and 9, respectively. Conclusions are given in Section 10. A number of tables showing the relevant statistics of the edit probabilities are displayed in the Appendices.

2. Description of Generators

2.1. *Description of Alternating Step Generator*

As shown in Fig. 1, the output of the ASG [11] is obtained by bitwise addition (modulo 2) of the output sequences of two binary LFSRs, LFSR_1 and LFSR_2, whose stop/go clocking

[1] The initialization scheme considered in [3] and [5] is more complicated than the one from the GSM standard.

[2] A sequence of independent uniformly distributed random variables over a finite set is called purely random.

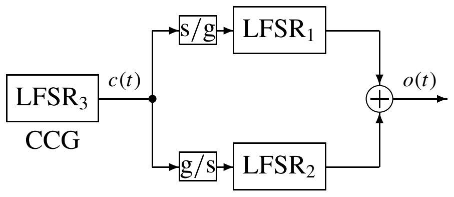

Fig. 1. Alternating step generator.

is defined by a binary clock-control generator (CCG), which is typically another, but regularly clocked LFSR, LFSR$_3$. It is assumed that LFSR$_1$ and LFSR$_2$ have different irreducible feedback polynomials of respective degrees r_1 and r_2 and coprime periods. At every step, only one LFSR is stepped and the output bit is assumed to be produced in the step-then-add manner. Let $c(t)$ denote the output bit of the CCG at step $t \geq 1$. In order to obtain the output bit $o(t)$ at step t, we first step LFSR$_1$ or LFSR$_2$ depending on whether $c(t) = 1$ or $c(t) = 0$, respectively, and then add modulo 2 the output bits of LFSR$_1$ and LFSR$_2$.

In [11] good standard cryptographic properties of the ASG, such as a long period, a high linear complexity, and approximately uniform relative frequency of short output patterns on a period are established, under the assumption that the clock-control sequence is a de Bruijn sequence and that the feedback polynomials of LFSR$_1$ and LFSR$_2$ are primitive. It is expected that similar results also hold if the CCG is an LFSR (LFSR$_3$) with a primitive feedback polynomial whose period is coprime to the periods of LFSR$_1$ and LFSR$_2$.

2.2. *Description of Bilateral Stop/Go Generator*

As shown in Fig. 2, the output of the BSGG is obtained by bitwise addition of the output sequences of two binary LFSRs, LFSR$_1$ and LFSR$_2$, which mutually clock-control each other by stop/go clocking (see [14] and [15]). It is assumed that LFSR$_1$ and LFSR$_2$ have primitive feedback polynomials of the same degree L. At each step, the output bit is assumed to be produced in the step-then-add manner as follows. Let $s_{i,L}(t)$ and $s_{i,L-1}(t)$, $i = 1, 2$, denote the contents at step $t \geq 0$ of the stages of LFSR$_i$ at positions L and $L - 1$, respectively. For any $t \geq 1$, the clock-control symbol $c(t)$, specifying which LFSRs are clocked in order to produce the output bit $o(t)$, is defined

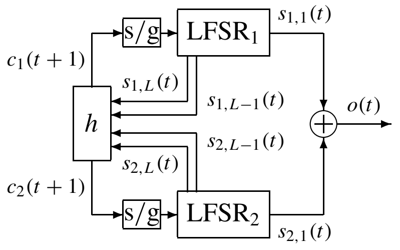

Fig. 2. Bilateral stop/go generator.

by $c(t) = h(s_{1,L}(t-1), s_{1,L-1}(t-1), s_{2,L}(t-1), s_{2,L-1}(t-1))$, where h is a 3-valued clock-control function of four binary variables such that $h(a,b,c,d)$ equals $\{1\}$ if $(a,b) = (0,1)$, $\{2\}$ if $(c,d) = (0,1) \neq (a,b)$, and $\{1,2\}$ otherwise. The output bit is produced as the binary (modulo 2) sum $o(t) = s_{1,1}(t) \oplus s_{2,1}(t)$. Note that the clock-control bits $c_i(t)$, $i = 1, 2$, are derived from $c(t)$ by using the stop/go clocking rule: at time t, LFSR$_i$ is clocked if $c_i(t) = 1$ and is not clocked if $c_i(t) = 0$.

Some theory of the BSGG is given in [14]. The state diagram of the BSGG consists of $3 \cdot 2^{L-2} - 1$ branched cycles each of length $T = 5 \cdot 2^{L-2} - 1$. At any cycle state there is at most one branch. Every branch has length 1 and starts with a state having no predecessor. By using L such that T is a prime, the linear complexity of the output sequence has a lower bound of the same order of magnitude as T (see [14] and [15]).

Interestingly, under the assumption that the regularly clocked LFSR sequences are independent and purely random, it turns out that the probability distribution of the 4-bit input to the clock-control function h is time-dependent. One can show that the underlying Markov chain with 16 states defined by these 4-bit inputs is ergodic, but, perhaps surprisingly, has a stationary probability distribution which is not uniform. As a consequence, $\Pr\{c(t) = \{1\}\}$ and $\Pr\{c(t) = \{2\}\}$ both quickly converge to $\frac{1}{5}$ as t increases. The stationary correlation coefficients between the first derivatives of the BSGG output sequence and of the output sequences of the stop/go clocked LFSR$_1$ and LFSR$_2$, respectively, are then both equal to $\frac{1}{5}$.

2.3. *Description of Alleged A5 Generator*

The alleged A5 keystream generator considered is shown in Fig. 3. The feedback polynomials of all three binary LFSRs are assumed to be primitive. Let $S_i(t) = (s_{i,l}(t))_{l=1}^{r_i}$ denote the internal state of LFSR$_i$ of length r_i at time $t \geq 0$ ($S_i(0)$ is the initial state) and let τ_i, $\tau_i \geq 1$, denote a middle tap from LFSR$_i$ used for clock-control, $i = 1, 2, 3$. For any $t \geq 1$, the clock-control symbol $c(t)$, specifying which LFSRs are clocked in order to produce the output bit $o(t)$, is defined by $c(t) = h(s_{1,\tau_1}(t-1), s_{2,\tau_2}(t-1), s_{3,\tau_3}(t-1))$, where h is a 4-valued majority function of three binary variables such that $h(s_1, s_2, s_3)$ is equal to $\{i, j\}$ if $s_i = s_j \neq s_k$, for $i < j$ and $k \neq i, j$, and to $\{1, 2, 3\}$ if $s_1 = s_2 = s_3$. The output bit is produced as the binary sum $o(t) = s_{1,1}(t) \oplus s_{2,1}(t) \oplus s_{3,1}(t)$. Note that the clock-control bits $c_i(t)$, $i = 1, 2, 3$, are derived from $c(t)$ by using the stop/go clocking rule.

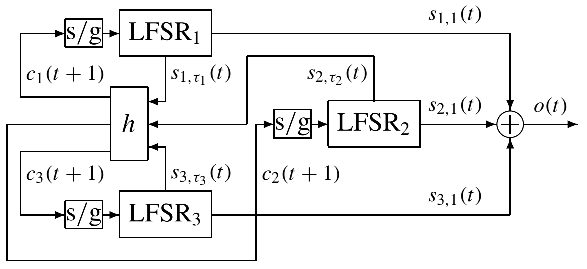

Fig. 3. Alleged A5 generator.

In [2] it is argued that the period of the output sequence is only slightly bigger than the period of the longest LFSR. Under the assumption that the regularly clocked LFSR sequences are independent and purely random, it is shown in [3] and [5] that the clock-control and output sequences are both purely random and, moreover, that the bitwise sum of any two stop/go clocked LFSR sequences is also purely random. This implies that each stop/go clocked LFSR sequence is statistically independent of the output sequence. Also, it follows that the probability distribution of the 3-bit input to the clock-control function h is uniform at any time.

3. Edit Probabilities for ASG

In this paper we use the notation $A = (a_i)_{i=1}^{\infty} = a_1, a_2, \ldots$ for a sequence of symbols from a finite set, and $A^n = (a_i)_{i=1}^{n} = a_1, a_2, \ldots, a_n$ for a string constituted by the first n symbols of A. Also, let $A_k^n = (a_i)_{i=k}^{n} = a_k, a_{k+1}, \ldots, a_n$. For a binary sequence A, its first (binary) derivative is denoted by $\dot{A} = \dot{a}_1, \dot{a}_2, \ldots$, where $\dot{a}_i = a_i \oplus a_{i+1}$. The bitwise binary complement of A is denoted by $\bar{A} = \overline{a_1}, \overline{a_2}, \ldots$, where $\overline{a_i} = 1 \oplus a_i$ is the binary complement of a_i.

For simplicity, we keep the same notation for random variables and their values.

3.1. *Edit Probability for One Input String*

Let $X^{n+2} = x_1, x_2, \ldots, x_{n+2}$ and $Y^{n+2} = y_1, y_2, \ldots, y_{n+2}$ denote two binary input strings and let $C^{n+1} = c_1, c_2, \ldots, c_{n+1}$ denote a binary clock-control string. Let $O^{n+1} = o_1, o_2, \ldots, o_{n+1} = F^{n+1}(X^{n+2}, Y^{n+2}, C^{n+1})$ denote the combination string of length $n+1$ produced from X^{n+2} and Y^{n+2} by the step-then-add alternating stepping according to C^{n+1} (X^{n+2} and Y^{n+2} correspond to the initial segments of regularly clocked LFSR$_1$ and LFSR$_2$ sequences, respectively). Accordingly, we initially have $o_1 = x_1 \oplus y_2$ if $c_1 = 0$ and $o_1 = x_2 \oplus y_1$ if $c_1 = 1$, whereas for any $0 \le s \le n$, if l denotes the number of ones in C^{s+1}, then $o_{s+1} = x_{l+1} \oplus y_{s+2-l}$.

The process of producing the first derivative of O^{n+1}, $\dot{O}^n = \dot{F}^n(X^{n+2}, Y^{n+2}, C^{n+1})$, from a given X^{n+2} is called the *edit transformation* of X^{n+2} into $\dot{O}^n$ according to Y^{n+2} and C^{n+1}.

Assume a probabilistic model where the strings X^{n+2}, Y^{n+2}, and C^{n+1} are independent and purely random. Let $Z^n = z_1, z_2, \ldots, z_n$ denote a binary output string. The *edit probability* for a given input string X^{n+2} and a given output string Z^n, denoted as $P(X^{n+2}; Z^n)$, is then defined as the probability that X^{n+2} is transformed into Z^n by the (random) edit transformation according to random Y^{n+2} and C^{n+1}. Formally, it is defined as the conditional probability

$$P(X^{n+2}; Z^n) = \Pr\{\dot{F}^n(X^{n+2}, Y^{n+2}, C^{n+1}) = Z^n \mid X^{n+2}\}. \tag{1}$$

The statistically optimal edit probability (minimizing the error probability when deciding on X^{n+2} given Z^n) is given as

$$\Pr\{X^{n+2}, \dot{F}^n(X^{n+2}, Y^{n+2}, C^{n+1}) = Z^n\} = P(X^{n+2}; Z^n) \cdot \Pr\{X^{n+2}\}. \tag{2}$$

Since $\Pr\{X^{n+2}\} = 2^{-n-2}$ for every X^{n+2}, the edit probability (1) is also statistically optimal.

If the input strings X^{n+2} and Y^{n+2} change places, then the resulting edit probability $P'(Y^{n+2}; Z^n)$ is not formally the same as the alternating stepping is not symmetric with respect to X^{n+2} and Y^{n+2} for a given clock-control string C^{n+1}. However, since $\dot{F}^n(X^{n+2}, Y^{n+2}, C^{n+1}) = \dot{F}^n(Y^{n+2}, X^{n+2}, \bar{C}^{n+1})$, it follows that $P' = P$.

Our basic objective is to examine whether the defined edit probability can be computed efficiently by an algorithm whose computational complexity is significantly smaller than 2^{2n+3}, which corresponds to the computation of (1) by the summation of the elementary probability 2^{-2n-3} over all Y^{n+2} and C^{n+1} such that $\dot{F}^n(X^{n+2}, Y^{n+2}, C^{n+1}) = Z^n$. To this end, for any $1 \le s \le n$ and $0 \le l \le s+1$, we define the *partial edit probability* $P(l, s)$ as the joint probability

$$P(l,s) = \Pr\{\dot{F}^s(X^{s+2}, Y^{s+2}, C^{s+1}) = Z^s, w(C^{s+1}) = l \mid X^{s+2}\}, \tag{3}$$

where $w(C^{s+1})$ is the Hamming weight of C^{s+1}. If $w(C^{s+1}) = l$, then $o_{s+1} = x_{l+1} \oplus y_{s+2-l}$, so that the edit transformation in (3) involves the prefixes X^{l+1} and Y^{s+2-l} only.

The next theorem enables the efficient computation of the edit probability based on a recursive property of the partial edit probability.

Theorem 1. *For any given X^{n+2} and Z^n, we have*

$$P(X^{n+2}; Z^n) = \sum_{l=0}^{n+1} P(l,n), \tag{4}$$

where the partial edit probability $P(l,n)$ is computed recursively by

$$P(l,s) = \tfrac{1}{4}P(l,s-1) + \tfrac{1}{2}\overline{\dot{x}_l \oplus z_s}\,P(l-1,s-1) \tag{5}$$

for $1 \le s \le n$ and $0 \le l \le s+1$, with the initial values $P(0,0) = P(1,0) = \frac{1}{2}$. (For each $0 \le s \le n$, if $l < 0$ or $l > s+1$, then it is assumed that $P(l,s) = 0$, so that the corresponding terms in (5) are not computed.)

Proof. First observe that (4) is an immediate consequence of the definition of the partial edit probability. Second, for $s = 1$, (3) implies that $P(0,1) = \frac{1}{8}$, $P(1,1) = \frac{1}{8} + \overline{\dot{x}_1 \oplus z_1}/4$, and $P(2,1) = \overline{\dot{x}_2 \oplus z_1}/4$. The same values can also be obtained by (5) from the given initial values.

Now assume that $s \ge 2$. We partition all clock-control strings C^{s+1} into two subsets depending on whether $c_{s+1} = 0$ or $c_{s+1} = 1$. For simplicity of notation, the conditioning on X^{s+2} is removed from (3) and the resulting equations. Then (3) can be put into the form

$$\begin{aligned} P(l,s) \;=\; & \Pr\{\dot{F}^s(X^{s+2}, Y^{s+2}, C^{s+1}) = Z^s, w(C^{s+1}) = l \mid c_{s+1} = 0\} \cdot \tfrac{1}{2} \\ & + \Pr\{\dot{F}^s(X^{s+2}, Y^{s+2}, C^{s+1}) = Z^s, w(C^{s+1}) = l \mid c_{s+1} = 1\} \cdot \tfrac{1}{2}. \end{aligned} \tag{6}$$

If $c_{s+1} = 0$, then $\dot{o}_s = \dot{y}_{s+1-l}$, and if $c_{s+1} = 1$, then $\dot{o}_s = \dot{x}_l$. Consequently,

$$\begin{aligned}
\Pr\{\dot{F}^s(X^{s+2}, Y^{s+2}, C^{s+1}) = Z^s, w(C^{s+1}) = l \mid c_{s+1} = 0\} \\
&= \Pr\{\dot{y}_{s+1-l} = z_s \mid \dot{F}^{s-1}(X^{s+1}, Y^{s+1}, C^s) = Z^{s-1}, w(C^s) = l\} \\
&\quad \cdot \Pr\{\dot{F}^{s-1}(X^{s+1}, Y^{s+1}, C^s) = Z^{s-1}, w(C^s) = l\} \qquad (7) \\
&= \Pr\{y_{s+2-l} = y_{s+1-l} \oplus z_s \mid \dot{F}^{s-1}(X^{s+1}, Y^{s+1}, C^s) = Z^{s-1}, w(C^s) = l\} \\
&\quad \cdot P(l, s-1) \qquad (8) \\
&= \tfrac{1}{2} \cdot P(l, s-1). \qquad (9)
\end{aligned}$$

Equation (9) follows from (8) because y_{s+2-l} remains independent of y_{s+1-l} when conditioned on $\dot{F}^{s-1}(X^{s+1}, Y^{s+1}, C^s) = Z^{s-1}$ and $w(C^s) = l$, as this condition involves only Y^{s+1-l} (not y_{s+2-l}).

Similarly,

$$\begin{aligned}
\Pr\{\dot{F}^s(X^{s+2}, Y^{s+2}, C^{s+1}) = Z^s, w(C^{s+1}) = l \mid c_{s+1} = 1\} \\
&= \Pr\{\dot{x}_l = z_s \mid \dot{F}^{s-1}(X^{s+1}, Y^{s+1}, C^s) = Z^{s-1}, w(C^s) = l-1\} \\
&\quad \cdot \Pr\{\dot{F}^{s-1}(X^{s+1}, Y^{s+1}, C^s) = Z^{s-1}, w(C^s) = l-1\} \\
&= \overline{\dot{x}_l \oplus z_s} \cdot P(l-1, s-1). \qquad (10)
\end{aligned}$$

Equation (5) directly follows from (6), (9), and (10). □

The time and space complexities of the recursive algorithm corresponding to Theorem 1 are clearly $O(n^2)$ and $O(n)$, respectively. The algorithm is thus feasible even if n is very large. The only computational problem is that the edit probability appears to be exponentially small in the string length, so that one has to deal with very small numbers. The following normalization turned out to be convenient in the conducted experiments: $\bar{P}(X^{n+2}; Z^n) = 2^{n+1}P(X^{n+2}; Z^n)$. It can be computed by the recursion (5) modified by mulitiplying its right-hand side and the initial values by 2.

3.2. *Edit Probability for Two Input Strings*

The *edit probability* for two given input strings X^{n+2} and Y^{n+2} and a given output string Z^n, denoted as $P(X^{n+2}, Y^{n+2}; Z^n)$, is defined as the probability that X^{n+2} and Y^{n+2} are transformed into Z^n by the (random) edit transformation according to a purely random C^{n+1}. The edit transformation is here defined as the process of producing $\dot{O}^n = \dot{F}^n(X^{n+2}, Y^{n+2}, C^{n+1})$ from given X^{n+2} and Y^{n+2}. More precisely, this edit probability is defined as

$$P(X^{n+2}, Y^{n+2}; Z^n) = \Pr\{\dot{F}^n(X^{n+2}, Y^{n+2}, C^{n+1}) = Z^n \mid X^{n+2}, Y^{n+2}\} \qquad (11)$$

(in contrast with (1), Y^{n+2} is here fixed rather than random).

By a similar argument as for $P(X^{n+2}; Z^n)$, it follows that this edit probability is statistically optimal in the probabilistic model where X^{n+2} is purely random. It is also statistically optimal in the probabilistic model where X^{n+2} and Y^{n+2} are independent and purely random (when deciding on X^{n+2} and Y^{n+2} given Z^n). This edit probability is symmetric with respect to the input strings X^{n+2} and Y^{n+2}, because $\dot{F}^n(X^{n+2}, Y^{n+2}, C^{n+1}) = \dot{F}^n(Y^{n+2}, X^{n+2}, \bar{C}^{n+1})$.

In a similar way as for the edit probability for one input string, one can then prove the following theorem.

Theorem 2. *For any given X^{n+2}, Y^{n+2}, and Z^n, we have*

$$P(X^{n+2}, Y^{n+2}; Z^n) = \sum_{l=0}^{n+1} P(l, n), \tag{12}$$

where the partial edit probability $P(l, n)$ is computed recursively by

$$P(l, s) = \tfrac{1}{2}\overline{\dot{y}_{s+1-l} \oplus z_s}\, P(l, s-1) + \tfrac{1}{2}\overline{\dot{x}_l \oplus z_s}\, P(l-1, s-1) \tag{13}$$

for $1 \le s \le n$ and $0 \le l \le s+1$, with the initial values $P(0,0) = P(1,0) = \frac{1}{2}$. (For each $0 \le s \le n$, if $l < 0$ or $l > s+1$, then it is assumed that $P(l, s) = 0$, so that the corresponding terms in (13) are not computed.)

The time and space complexities of the recursive computation are also $O(n^2)$ and $O(n)$, respectively, and the normalization can be performed in the same way as for the edit probability for one input string.

The edit probability for one input string may be used to obtain a relatively small number of candidates for the initial state of each of the stop/go clocked LFSRs from a given output sequence. However, given a number of candidate initial states for one of the LFSRs, it is more efficient to recover the initial state of the other one by using the edit probability for two input strings.

4. Edit Probability for BSGG

Let $X^{n+2} = x_1, x_2, \dots, x_{n+2}$ and $Y^{n+2} = y_1, y_2, \dots, y_{n+2}$, denote two binary input sequences and let $C^{n+1} = c_1, c_2, \dots, c_{n+1}$ denote a 3-valued clock-control string, where $c_i \in \mathcal{C}$, $\mathcal{C} = \{\{1\}, \{2\}, \{1,2\}\}$. Let $O^{n+1} = o_1, o_2, \dots, o_{n+1} = F^{n+1}(X^{n+2}, Y^{n+2}, C^{n+1})$ denote the combination string produced from X^{n+2} and Y^{n+2} by the bilateral stop/go clocking according to C^{n+1}, where X^{n+2} and Y^{n+2} correspond to the initial segments of regularly clocked LFSR$_1$ and LFSR$_2$ sequences, respectively.

More precisely, we initially have $o_1 = x_2 \oplus y_1$ if $c_1 = \{1\}$, $o_1 = x_1 \oplus y_2$ if $c_1 = \{2\}$, and $o_1 = x_2 \oplus y_2$ if $c_1 = \{1, 2\}$. Let $w_i(C^{s+1})$ denote the number of occurrences of the symbol $\{i\}$, $i = 1, 2$, in a string C^{s+1}. For simplicity, let $w_1(C^{s+1}) = l_1$ and $w_2(C^{s+1}) = l_2$. The number of occurrences of the symbol $\{1, 2\}$ in C^{s+1} is then $s + 1 - l_1 - l_2$. Then for any $0 \le s \le n$, $o_{s+1} = x_{s+2-l_2} \oplus y_{s+2-l_1}$. Further, the clock-control string is generated according to the BSGG scheme from Fig. 2 as a function $C^{n+1} = \mathcal{H}^{n+1}(X^{n+L}, Y^{n+L})$. Namely, $c_1 = h(x_L, x_{L-1}, y_L, y_{L-1})$ and for $0 \le s \le n$, $c_{s+2} = h(x_{L+s+1-l_2}, x_{L+s-l_2}, y_{L+s+1-l_1}, y_{L+s-l_1})$. Altogether, we can write $O^{n+1} = \mathcal{G}^{n+1}(X^{n+L'}, Y^{n+L'})$, where $L' = \max(L, 2)$.

We adopt an approximative model for the BSGG which allows us to define and recursively compute a suitable edit probability. Introduce an auxiliary random binary string $R^{2n+2} = r_1, r_2, \dots, r_{2n+2}$ in order to replace the input string Y^{n+L} in the role of generating the input bits for the clock-control function h. In the approximative BSGG model the clock-control string C^{n+1} is generated as follows. Initially, we have $c_1 = h(x_L, x_{L-1}, r_2, r_1)$ and for $0 \le s \le n$, $c_{s+2} = h(x_{L+s+1-l_2}, x_{L+s-l_2}, r_{2s+4}, r_{2s+3})$. We represent this by $C^{n+1} = H^{n+1}(X^{n+L}, R^{2n+2})$. The output string is generated as $O^{n+1} = F^{n+1}(X^{n+2}, Y^{n+2}, C^{n+1})$. Altogether, we can write $O^{n+1} = G^{n+1}(X^{n+L'}, Y^{n+2}, R^{2n+2})$.

The process of producing the first derivative of O^{n+1}, $\dot{O}^n = \dot{G}^n(X^{n+L'}, Y^{n+2}, R^{2n+2})$, from a given input string $X^{n+L'}$ according to an input string Y^{n+2} and an auxiliary clock-control string R^{2n+2} is called the *edit transformation* of $X^{n+L'}$ into $\dot{O}^n$ according to Y^{n+2} and R^{2n+2}.

Assume a probabilistic model where the strings $X^{n+L'}$, Y^{n+2}, and R^{2n+2} are independent and purely random. Alternatively, R^{2n+2} can be more precisely modeled as a sequence of independent pairs of successive bits (r_{2s+1}, r_{2s+2}), $0 \le s \le n$, where the probability distribution of each pair is derived from the (nonuniform) stationary distribution of the Markov chain associated with the 4-bit inputs to the clock-control function. However, it turns out that the simplified model is a sufficiently good approximation. Let $Z^n = z_1, z_2, \dots, z_n$ denote a given output string. The *edit probability* for a given input string $X^{n+L'}$ and a given output string Z^n is the probability that $X^{n+L'}$ is transformed into Z^n by the (random) edit transformation according to random Y^{n+2} and R^{2n+2}. Formally, it is defined as the conditional probability

$$P(X^{n+L'}; Z^n) = \Pr\{\dot{G}^n(X^{n+L'}, Y^{n+2}, R^{2n+2}) = Z^n \mid X^{n+L'}\}. \tag{14}$$

The statistically optimal edit probability (minimizing the error probability when deciding on $X^{n+L'}$ given Z^n) is given as

$$\Pr\{X^{n+L'}, \dot{G}^n(X^{n+L'}, Y^{n+2}, R^{2n+2}) = Z^n\} = P(X^{n+L'}; Z^n) \cdot \Pr\{X^{n+L'}\}. \tag{15}$$

As $\Pr\{X^{n+L'}\} = 2^{-(n+L')}$, the edit probability (14) is also statistically optimal.

Our objective is to examine whether the defined edit probability can be computed efficiently by a recursive algorithm whose computational complexity is significantly smaller than $O(2^{3n+4})$, which corresponds to the computation of (14) by the summation of the elementary probability $2^{-(3n+4)}$ over all Y^{n+2} and R^{2n+2} such that $\dot{G}^n(X^{n+L'}, Y^{n+2}, R^{2n+2}) = Z^n$. To this end, we define the *partial edit probability* depending on the distribution of symbols in the clock-control string C^{n+1}.

For any $0 \le s \le n$, a pair (l_1, l_2) is said to be *permissible* if $0 \le l_1, l_2 \le s+1$ and $l_1 + l_2 \le s+1$. For a given s, the set of all the permissible values of (l_1, l_2) is denoted by $\mathcal{L}_s$. For any $1 \le s \le n$ and $(l_1, l_2) \in \mathcal{L}_s$, the partial edit probability is defined as the conditional joint probability

$$P(l_1, l_2, s) = \Pr\{\dot{O}^s = Z^s, w_1(C^{s+1}) = l_1, w_2(C^{s+1}) = l_2 \mid X^{s+L'}\}, \tag{16}$$

where $\dot{O}^s = \dot{F}^s(X^{s+2}, Y^{s+2}, C^{s+1})$ and $C^{s+1} = H^{s+1}(X^{s+L}, R^{2s+2})$. The following theorem shows how to compute the edit probability efficiently, on the basis of a recursive property of the partial edit probability.

Theorem 3. *For any given $X^{n+L'}$ and Z^n, we have*

$$P(X^{n+L'}; Z^n) = \sum_{(l_1, l_2) \in \mathcal{L}_n} P(l_1, l_2, n), \tag{17}$$

where the partial edit probability $P(l_1, l_2, n)$ *is computed recursively by*

$$\begin{aligned} P(l_1, l_2, s) \;=\; & P(l_1 - 1, l_2, s - 1)\overline{z_s \oplus \dot{x}_{s+1-l_2}}\,\overline{x_{L+s-l_2}}x_{L+s-l_2-1} \\ & + \tfrac{1}{8} P(l_1, l_2 - 1, s - 1)\overline{\overline{x_{L+s-l_2+1}}x_{L+s-l_2}} \\ & + \tfrac{3}{8} P(l_1, l_2, s - 1)\overline{\overline{x_{L+s-l_2}}x_{L+s-l_2-1}} \end{aligned} \tag{18}$$

for $1 \le s \le n$ *and all* $(l_1, l_2) \in \mathcal{L}_s$, *with the initial values* $P(0, 0, 0) = \frac{3}{4}\overline{\overline{x_L}x_{L-1}}$, $P(0, 1, 0) = \frac{1}{4}\overline{\overline{x_L}x_{L-1}}$, *and* $P(1, 0, 0) = \overline{x_L}x_{L-1}$. (*For each* $0 \le s \le n$, *if* (l_1, l_2) *is not permissible*, *then it is assumed that* $P(l_1, l_2, s) = 0$, *so that the corresponding terms in* (18) *are not computed*.)

Proof. First observe that (17) is a direct consequence of (16) and (14).

Assume that $s \ge 2$. We partition all clock-control strings C^{s+1} into three subsets with respect to the value of the last symbol c_{s+1}. For simplicity of notation, the conditioning on $X^{s+L'}$ is removed from (16) and the resulting equations. Then (16) can be put into the form

$$\begin{aligned} & P(l_1, l_2, s) \\ & \quad = \Pr\{\dot{o}_s = z_s \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1 - 1, w_2(C^s) = l_2, c_{s+1} = \{1\}\} \\ & \qquad \cdot \Pr\{c_{s+1} = \{1\} \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1 - 1, w_2(C^s) = l_2\} \\ & \qquad \cdot \Pr\{\dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1 - 1, w_2(C^s) = l_2\} \\ & \qquad + \Pr\{\dot{o}_s = z_s \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1, w_2(C^s) = l_2 - 1, c_{s+1} = \{2\}\} \\ & \qquad \cdot \Pr\{c_{s+1} = \{2\} \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1, w_2(C^s) = l_2 - 1\} \\ & \qquad \cdot \Pr\{\dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1, w_2(C^s) = l_2 - 1\} \\ & \qquad + \Pr\{\dot{o}_s = z_s \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1, w_2(C^s) = l_2, c_{s+1} = \{1, 2\}\} \\ & \qquad \cdot \Pr\{c_{s+1} = \{1, 2\} \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1, w_2(C^s) = l_2\} \\ & \qquad \cdot \Pr\{\dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1, w_2(C^s) = l_2\}. \end{aligned} \tag{19}$$

The third factor in each addend of (19) is easily recognized to be the partial edit probability appearing in the corresponding addend of (18) (use (16) for $s - 1$).

Now, under the condition that $w_1(C^s) = l_1 - 1$ and $w_2(C^s) = l_2$, we have $c_{s+1} = h(x_{L+s-l_2}, x_{L+s-l_2-1}, r_{2s+2}, r_{2s+1})$, which equals $\{1\}$ if and only if $(x_{L+s-l_2}, x_{L+s-l_2-1}) = (0, 1)$. Further, if $c_{s+1} = \{1\}$, then $\dot{o}_s = \dot{x}_{s+1-l_2}$. Similarly, under the condition that $w_1(C^s) = l_1$ and $w_2(C^s) = l_2 - 1$, we have $c_{s+1} = h(x_{L+s-l_2+1}, x_{L+s-l_2}, r_{2s+2}, r_{2s+1})$, which equals $\{2\}$ if and only if $(x_{L+s-l_2+1}, x_{L+s-l_2}) \ne (0, 1)$ and $(r_{2s+2}, r_{2s+1}) = (0, 1)$. Also, if $c_{s+1} = \{2\}$, then $\dot{o}_s = \dot{y}_{s+1-l_1}$. Finally, under the condition that $w_1(C^s) = l_1$ and $w_2(C^s) = l_2$, we have $c_{s+1} = h(x_{L+s-l_2+1}, x_{L+s-l_2}, r_{2s+2}, r_{2s+1})$, which equals $\{1, 2\}$ if and only if $(x_{L+s-l_2}, x_{L+s-l_2-1}) \ne (0, 1)$ and $(r_{2s+2}, r_{2s+1}) \ne (0, 1)$. Also, if $c_{s+1} = \{1, 2\}$, then $\dot{o}_s = \dot{x}_{s+1-l_2} \oplus \dot{y}_{s+1-l_1}$.

Consequently, we have (conditioned on $X^{s+L'}$) that

$$\begin{aligned}\Pr\{c_{s+1} = \{1\} \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1 - 1, w_2(C^s) = l_2\} \\ = \overline{x_{L+s-l_2}} x_{L+s-l_2-1}, \qquad (20)\\ \Pr\{c_{s+1} = \{2\} \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1, w_2(C^s) = l_2 - 1\} \\ = \overline{\overline{x_{L+s-l_2+1}} x_{L+s-l_2}} \cdot \tfrac{1}{4}, \qquad (21)\\ \Pr\{c_{s+1} = \{1, 2\} \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1, w_2(C^s) = l_2\} \\ = \overline{\overline{x_{L+s-l_2}} x_{L+s-l_2-1}} \cdot \tfrac{3}{4}. \qquad (22)\end{aligned}$$

Further, we get

$$\begin{aligned}\Pr\{\dot{o}_s = z_s \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1 - 1, w_2(C^s) = l_2, c_{s+1} = \{1\}\} \\ = \overline{z_s \oplus \dot{x}_{s+1-l_2}}, \qquad (23)\\ \Pr\{\dot{o}_s = z_s \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1, w_2(C^s) = l_2 - 1, c_{s+1} = \{2\}\} = \tfrac{1}{2}, \qquad (24)\\ \Pr\{\dot{o}_s = z_s \mid \dot{O}^{s-1} = Z^{s-1}, w_1(C^s) = l_1, w_2(C^s) = l_2, c_{s+1} = \{1, 2\}\} = \tfrac{1}{2}. \qquad (25)\end{aligned}$$

Equation (24) follows from $\dot{o}_s = \dot{y}_{s+1-l_1}$ by taking into account that y_{s+2-l_1} remains independent of y_{s+1-l_1} when conditioned on $\dot{O}^{s-1} = Z^{s-1}$, $w_1(C^s) = l_1$, and $w_2(C^s) = l_2 - 1$, as this condition involves only Y^{s+1-l_1} (not y_{s+2-l_1}). Equation (25) is proved analogously.

Equation (18) is obtained from (19) by plugging in the determined probabilities.

For $s = 1$, the edit probability values are directly obtained from (16). When these values are expressed in terms of the unknown initial values by the recursion (18), a system of linear equations is formed. The initial values are then determined by solving this system. □

The time and space complexities of the recursive algorithm corresponding to Theorem 3 are $O(n^3)$ and $O(n^2)$, respectively. Since the edit probability is exponentially small in the string length, the following normalization turns out to be computationally convenient: $\bar{P}(X^{n+L'}; Z^n) = 2^{n+1} P(X^{n+L'}; Z^n)$. It can be computed by the recursion (18) modified by mulitiplying its right-hand side and the initial values by 2.

Note that one can similarly define and compute the edit probability with respect to LFSR$_2$ instead of LFSR$_1$ as a target shift register.

5. Edit Probability for Alleged A5

Let $X^{n+2} = x_1, x_2, \ldots, x_{n+2}$, $Y^{n+2} = y_1, y_2, \ldots, y_{n+2}$, and $U^{n+2} = u_1, u_2, \ldots, u_{n+2}$ denote three binary input strings and let $C^{n+1} = c_1, c_2, \ldots, c_{n+1}$ denote a 4-valued clock-control string, where $c_i \in \mathcal{C}$, $\mathcal{C} = \{\{1, 2\}, \{2, 3\}, \{1, 3\}, \{1, 2, 3\}\}$. Let $O^{n+1} = o_1, o_2, \ldots, o_{n+1} = F^{n+1}(X^{n+2}, Y^{n+2}, U^{n+2}, C^{n+1})$ denote the combination string produced from X^{n+2}, Y^{n+2}, and U^{n+2} by the alleged A5 stop/go clocking according to C^{n+1}, where X^{n+2}, Y^{n+2}, and U^{n+2} correspond to the initial segments of regularly clocked LFSR$_1$, LFSR$_2$, and LFSR$_3$ sequences, respectively, and C^{n+1} is generated independently.

More precisely, we initially have $o_1 = x_2 \oplus y_2 \oplus u_1$ if $c_1 = \{1, 2\}$, $o_1 = x_1 \oplus y_2 \oplus u_2$ if $c_1 = \{2, 3\}$, $o_1 = x_2 \oplus y_1 \oplus u_2$ if $c_1 = \{1, 3\}$, and $o_1 = x_2 \oplus y_2 \oplus u_2$ if $c_1 = \{1, 2, 3\}$. Let $w_{ij}(C^{s+1})$ denote the number of occurrences of the symbol $\{i, j\}$ in C^{s+1}, $1 \leq i < j \leq 3$. For simplicity, let $w_{12}(C^{s+1}) = l_1$, $w_{23}(C^{s+1}) = l_2$, and $w_{13}(C^{s+1}) = l_3$. The number of occurrences of the symbol $\{1, 2, 3\}$ in C^{s+1} is then $s + 1 - l_1 - l_2 - l_3$. Then for any $0 \leq s \leq n$, $o_{s+1} = x_{s+2-l_2} \oplus y_{s+2-l_3} \oplus u_{s+2-l_1}$.

The process of producing the first derivative of O^{n+1}, $\dot{O}^n = \dot{F}^n(X^{n+2}, Y^{n+2}, U^{n+2}, C^{n+1})$, is called an edit transformation of X^{n+2}, Y^{n+2}, and U^{n+2} into $\dot{O}^n$ according to C^{n+1}.

In the alleged A5 keystream generator, instead of being independent of the LFSR strings, the clock-control string is produced as a function of their phase-shifted versions. A more realistic edit transformation should incorporate this feature. To achieve a divide-and-conquer effect, the edit transformation should involve exactly two input strings, in view of the fact that just one input string cannot be recovered from the output string. This is because the bitwise sum of any two stop/go clocked LFSR sequences is purely random (under the assumption that the regularly clocked LFSR sequences are independent and purely random).

Let $\tau_i' = \max(\tau_i, 2)$, where τ_i is the tap position in LFSR$_i$ used for clock-control, $i = 1, 2, 3$. Then the combination string $O^{n+1} = F^{n+1}(X^{n+2}, Y^{n+2}, U^{n+2}, C^{n+1})$ is produced in the same way as above, whereas the clock-control string C^{n+1} is generated as a function $\mathcal{H}^{n+1}(X_{\tau_1}^{n+\tau_1}, Y_{\tau_2}^{n+\tau_2}, U_{\tau_3}^{n+\tau_3})$. More precisely, $c_1 = h(x_{\tau_1}, y_{\tau_2}, u_{\tau_3})$ and for any $0 \leq s \leq n$, $c_{s+2} = h(x_{s+1+\tau_1-l_2}, y_{s+1+\tau_2-l_3}, u_{s+1+\tau_3-l_1})$. Altogether, we can write $O^{n+1} = \mathcal{G}^{n+1}(X^{n+\tau_1'}, Y^{n+\tau_2'}, U^{n+\tau_3'})$.

In order to define an edit probability for two input strings that can be computed recursively, we adopt an approximative model for the alleged A5 where instead of $U_{\tau_3}^{n+\tau_3}$, an auxiliary random binary string R^{n+1} is used to produce C^{n+1}. Thus, the combination string O^{n+1} is produced as $F^{n+1}(X^{n+2}, Y^{n+2}, U^{n+2}, C^{n+1})$ and the clock-control string C^{n+1} is generated as a function $H^{n+1}(X_{\tau_1}^{n+\tau_1}, Y_{\tau_2}^{n+\tau_2}, R^{n+1})$. More precisely, $c_1 = h(x_{\tau_1}, y_{\tau_2}, r_1)$ and for any $0 \leq s \leq n$, $c_{s+2} = h(x_{s+1+\tau_1-l_2}, y_{s+1+\tau_2-l_3}, r_{s+2})$. Altogether, we have $O^{n+1} = G^{n+1}(X^{n+\tau_1'}, Y^{n+\tau_2'}, U^{n+2}, R^{n+1})$.

The process of producing the first derivative of O^{n+1}, $\dot{O}^n = \dot{G}^n(X^{n+\tau_1'}, Y^{n+\tau_2'}, U^{n+2}, R^{n+1})$, from given input strings $X^{n+\tau_1'}$ and $Y^{n+\tau_2'}$ according to an input string U^{n+2} and an auxiliary clock-control string R^{n+1} is called the *edit transformation* of $X^{n+\tau_1'}$ and $Y^{n+\tau_2'}$ into $\dot{O}^n$ according to U^{n+2} and R^{n+1}.

Assume a probabilistic model where the strings $X^{n+\tau_1'}$, $Y^{n+\tau_2'}$, U^{n+2}, and R^{n+1} are independent and purely random. Note that this model of R^{n+1} is in accordance with the uniform probability distribution of the 3-bit input to the clock-control function (unlike the BSGG). Let $Z^n = z_1, z_2, \ldots, z_n$ denote a binary output string. The *edit probability* for given input strings $X^{n+\tau_1'}$ and $Y^{n+\tau_2'}$ and a given output string Z^n is then defined as the probability that $X^{n+\tau_1'}$ and $Y^{n+\tau_2'}$ are transformed into Z^n by the (random) edit transformation according to random U^{n+2} and R^{n+1}. Formally, it is defined as the conditional probability

$$P(X^{n+\tau_1'}, Y^{n+\tau_2'}; Z^n)$$
$$= \Pr\{\dot{G}^n(X^{n+\tau_1'}, Y^{n+\tau_2'}, U^{n+2}, R^{n+1}) = Z^n \mid X^{n+\tau_1'}, Y^{n+\tau_2'}\}. \tag{26}$$

It is symmetric with respect to $X^{n+\tau_1'}$ and $Y^{n+\tau_2'}$.

The statistically optimal edit probability (minimizing the error probability when deciding on $X^{n+\tau_1'}$ and $Y^{n+\tau_2'}$ given Z^n) is given as

$$\Pr\{X^{n+\tau_1'}, Y^{n+\tau_2'}, \dot{G}^n(X^{n+\tau_1'}, Y^{n+\tau_2'}, U^{n+2}, R^{n+1}) = Z^n\}$$
$$= P(X^{n+\tau_1'}, Y^{n+\tau_2'}; Z^n) \cdot \Pr\{X^{n+\tau_1'}, Y^{n+\tau_2'}\}. \tag{27}$$

As $\Pr\{X^{n+\tau_1'}, Y^{n+\tau_2'}\} = 2^{-(2n+\tau_1'+\tau_2')}$, the edit probability (26) is also statistically optimal.

For any $0 \le s \le n$, a triple (l_1, l_2, l_3) is said to be *permissible* if $0 \le l_1, l_2, l_3 \le s+1$ and $l_1 + l_2 + l_3 \le s+1$. For a given s, the set of all the permissible values of (l_1, l_2, l_3) is denoted by $\mathcal{L}_s$. For any $1 \le s \le n$ and $(l_1, l_2, l_3) \in \mathcal{L}_s$, the corresponding partial edit probability is defined as

$$P(l_1, l_2, l_3, s) = \Pr\{\dot{O}^s = Z^s, w_{12}(C^{s+1}) = l_1, w_{23}(C^{s+1}) = l_2,$$
$$w_{13}(C^{s+1}) = l_3 \mid X^{n+\tau_1'}, Y^{n+\tau_2'}\}, \tag{28}$$

where $\dot{O}^s = \dot{F}^s(X^{s+2}, Y^{s+2}, U^{s+2}, C^{s+1})$ and $C^{s+1} = H^{s+1}(X_{\tau_1}^{s+\tau_1}, Y_{\tau_2}^{s+\tau_2}, R^{s+1})$. Its recursive computation is established by the following theorem which can be proved by using a similar, although more involved, technique as Theorem 3.

Theorem 4. *For any given $X^{n+\tau_1'}$, $Y^{n+\tau_2'}$, and Z^n, we have*

$$P(X^{n+\tau_1'}, Y^{n+\tau_2'}; Z^n) = \sum_{(l_1,l_2,l_3)\in\mathcal{L}_n} P(l_1, l_2, l_3, n), \tag{29}$$

where the partial edit probability $P(l_1, l_2, l_3, n)$ is computed recursively by

$$P(l_1, l_2, l_3, s) = \tfrac{1}{2} P(l_1 - 1, l_2, l_3, s-1)\overline{z_s \oplus \dot{x}_{s+1-l_2} \oplus \dot{y}_{s+1-l_3}}\; \overline{x_{s+\tau_1-l_2} \oplus y_{s+\tau_2-l_3}}$$
$$+ \tfrac{1}{4}(P(l_1, l_2 - 1, l_3, s-1)(x_{s+1+\tau_1-l_2} \oplus y_{s+\tau_2-l_3})$$
$$+ P(l_1, l_2, l_3 - 1, s-1)(x_{s+\tau_1-l_2} \oplus y_{s+1+\tau_2-l_3})$$
$$+ P(l_1, l_2, l_3, s-1)\overline{x_{s+\tau_1-l_2} \oplus y_{s+\tau_2-l_3}}) \tag{30}$$

for $1 \le s \le n$ and all $(l_1, l_2, l_3) \in \mathcal{L}_s$, with the initial values $P(0,0,0,0) = P(1,0,0,0) = \overline{x_{\tau_1} \oplus y_{\tau_2}}/2$ and $P(0,1,0,0) = P(0,0,1,0) = (x_{\tau_1} \oplus y_{\tau_2})/2$. (For each $0 \le s \le n$, if (l_1, l_2, l_3) is not permissible, then it is assumed that $P(l_1, l_2, l_3, s) = 0$, so that the corresponding terms in (30) *are not computed.)*

The time and space complexities of the recursive algorithm corresponding to Theorem 4 are $O(n^4)$ and $O(n^3)$, respectively.

Since the edit probability is exponentially small in the string length, the following normalization turned out to be computationally convenient in the conducted experiments: $\bar{P}(X^{n+\tau_1'}, Y^{n+\tau_2'}; Z^n) = 2^{n+1} P(X^{n+\tau_1'}, Y^{n+\tau_2'}; Z^n)$. It can be computed by the recursion (30) modified by mulitiplying its right-hand side and the initial values by 2.

6. Correlation Attacks: General Framework

It is assumed that the LFSR feedback polynomials and a sufficiently long segment of the keystream sequence, in the known plaintext scenario, are known. The objective of cryptanalysis is to reconstruct the secret-key-dependent LFSR initial states by a method faster than exhaustive search.

6.1. *Main Lines*

The correlation attack goes on as follows. Depending on the keystream generator considered, pick a target subset of the LFSRs whose initial states are to be reconstructed. Let r be the total length of the chosen LFSRs. Initially, generate the output string of appropriate length n as the first derivative of a given segment of the keystream sequence (the keystream segment length is $n+1$). Then assume the unknown initial states of the chosen subset of the LFSRs and generate input strings of appropriate lengths as the initial segments of the corresponding regularly clocked LFSR sequences. In Section 6.3 we argue that the length n should be proportional to r, whereas the lengths of the input strings are defined as the maximum possible lengths that may give rise to an output string of length n (depending on the generator). Compute the (normalized) edit probability associated with the generated input strings and the output string by using the respective recursive algorithm. Repeat this for every possible combination of the chosen LFSR initial states, altogether 2^r of them. The complexity is thus 2^r steps each consisting of the computation of the edit probability which itself has complexity polynomial in n, depending on the keystream generator.

Provided that the edit probability is sufficiently larger in the case when our guess about the unknown input strings is correct than in the opposite case, the candidates for the correct combination of the LFSR initial states, roughly speaking, can be obtained as those with the computed edit probability close to being maximal. In order to treat this issue more precisely, we introduce a suitable statistical hypothesis testing model.

6.2. *Statistical Hypothesis Testing*

Consider the probability distribution of the edit probability under the following two hypotheses:

- $\mathbf{H_0}$ (*correlated case*): The output string is produced by the respective keystream generator in which the LFSR strings are assumed to be purely random and independent.
- $\mathbf{H_1}$ (*independent case*): The output string and the input strings are purely random and independent.

Clearly, $\mathbf{H_0}$ models the case when the guessed input string combination is correct, and $\mathbf{H_1}$ models the opposite case.

For the maximum edit probability decision rule to work, it is necessary that the separation between the probability distributions in the correlated and independent cases increases with the length n of the output string, and the faster the increase, the smaller the string length required for successful decision making. Since a theoretical analysis of the separation between the two probability distributions appears to be very difficult, the separation should be measured experimentally.

As we deal with a decision making problem, the separation is measured by the false alarm probability (derived from $\mathbf{H_1}$) when the missing event probability (derived from $\mathbf{H_0}$) is fixed. Since the number of correct input string combinations is only one, the missing event probability can be fixed to a value which need not be very small (e.g., $p_\mathrm{m} = 0.1$ or $p_\mathrm{m} = 0.01$). Then a threshold is set according to p_m and a tested input string combination is classified under $\mathbf{H_0}$ or $\mathbf{H_1}$ depending on whether the edit probability is bigger or smaller than the threshold. Thus, the false alarm probability p_f becomes a function of n, and if and only if this function is decreasing, then the separation between the two distributions increases with n, as desired.

In the correlation attack, the threshold as a function of n is estimated empirically, by computer simulations. A tested input string combination is classified as a candidate if the corresponding edit probability is bigger than or equal to the threshold.

6.3. *Output String Length*

Ideally, if n is large enough, then there will remain only one candidate for the correct input string combination. This can happen only if the false alarm probability $p_\mathrm{f}(n)$ is sufficiently small. Namely, since the expected number of false candidates is $(2^r-1)\,p_\mathrm{f}(n)$, for an average output string, the correlation attack is deemed successful if and only if, approximately,

$$2^r\,p_\mathrm{f}(n) \leq 1. \tag{31}$$

The false alarm probability as a function of n is estimated empirically by computer simulations. If the decrease of $p_\mathrm{f}(n)$, for large n, is consistent with the exponential form ab^n, where $b < 1$, then the parameters a and b can be estimated by the least mean square approximation method applied to the logarithms (to the base 2) of the false alarm probability estimates. Then (31) reduces to

$$n \geq \frac{r + \log_2 a}{-\log_2 b}, \tag{32}$$

which means that the required output string length is essentially linear in the total length of the chosen subset of the LFSRs. This is in accordance with the capacity argument for the communication channel that can be associated with the reconstruction problem.

6.4. *Final Reconstruction*

The number of candidate input string combinations obtained cannot be reduced to just one by increasing the length n if the threshold is computed according to a given p_m, because the targeted LFSR initial states generating relatively small (positive or negative) phase shifts of the correct input strings give rise to the edit probability values that are also close to being maximal. For all the keystream generators considered, this can be explained by the respective recursions for the partial edit probability which change only in the beginning for a negative phase shift (delay) or at the end for a positive phase shift, and remain the same for the best part of the input strings. Therefore, small phase shifts of the correct input strings are not well modeled by the hypothesis $\mathbf{H_1}$ (independent case).

Accordingly, a relatively small number of candidate initial states for the chosen subset of the LFSRs are obtained.

They are ranked and further tested in order of decreasing edit probabilities. The correct initial states, along with the correct initial states of the remaining LFSRs, are identified in the final stage of the attack. Namely, for each candidate initial state combination, all possible initial states of the remaining LFSRs are tested by generating the output sequence and comparing it with the known keystream sequence. If no match is found, then the candidate initial state combination is discarded. The solution for the targeted LFSR initial states need not be unique if the keystream generator (e.g., the alleged A5) is such that there exist equivalent initial states, producing the same output sequence.

If r' is the total length of the remaining LFSRs, then the time complexity of the final stage is $O(2^{r'})$. If $r' > r$ (for the ASG and the alleged A5), then the time complexity of the final stage can be reduced by a more sophisticated method based on the appropriate edit distance (see [6] for the ASG).

7. Correlation Attack on ASG

The general lines of the correlation attack are described in Section 6. Here we specify the details adapted to the ASG and provide experimental results obtained by computer simulations.

The objective is to recover the initial state of LFSR_1 or LFSR_2 individually, by using the (normalized) edit probability for one input string introduced in Section 3.1. A number of candidates for the initial state of LFSR_i are obtained in this way. Then the candidates for the initial state of the other shift register LFSR_j, $j \neq i$, can be obtained either by using the edit probability for one input string or the edit probability for two input strings defined in Section 3.2. In the final stage, the correct initial states of LFSR_1, LFSR_2, and LFSR_3 are all reconstructed. In view of the cyclic state diagram of the ASG, one may expect that the solution for the LFSR initial states is unique.

7.1. *Statistical Discrimination and Output String Length*

The normalized edit probability $\bar{P}(X^{n+2}; Z^n)$ is computed for the output string Z^n and an input string X^{n+2}, where Z^n is constituted by the first n bits of the first derivative of the known keystream sequence and X^{n+2} is the first $n+2$ output bits generated by the LFSR_i recursion from an assumed initial state. The probability distribution of $\bar{P}(X^{n+2}; Z^n)$ is considered under the following two hypotheses:

- $\mathbf{H_0}$ (*correlated case*): X^{n+2}, Y^{n+2}, and C^{n+1} are purely random and independent and $Z^n = \dot{F}^n(X^{n+2}, Y^{n+2}, C^{n+1})$.
- $\mathbf{H_1}$ (*independent case*): X^{n+2} and Z^n are purely random and independent.

To measure the separation between $\mathbf{H_0}$ and $\mathbf{H_1}$, we conducted systematic experiments and produced histograms of the two distributions for each $n = 100, (10), 1000$ on random samples generated from 10,000 pairs (X^{n+2}, Z^n) according to $\mathbf{H_0}$ and $\mathbf{H_1}$, respectively. They show that although the distributions cannot be well approximated by the normal distributions, the separation becomes significant even for relatively small values of n. It thus turns out that $\bar{P}$ is much larger under $\mathbf{H_0}$ than under $\mathbf{H_1}$, for a sufficiently large n.

Table 1. ASG: Estimation of a and b on 20*, 40**, and 60*** points.

p_m	a^*	a^{**}	a^{***}	b^*	b^{**}	b^{***}
0.1	0.7815	—	—	0.9799	—	—
0.01	1.288	1.922	—	0.9879	0.9858	—
0.001	1.346	2.184	2.897	0.9926	0.9901	0.9891

For illustration, Tables 1A and 2A given in Appendix A display the observed minimum, maximum, mean, and median values along with the standard deviation of $\bar{P}$ for each $n = 100, (100), 800$ under $\mathbf{H_1}$ and $\mathbf{H_0}$, respectively.

For $p_m = 0.1$, $p_m = 0.01$, and $p_m = 0.001$, Table 3A given in Appendix A displays the estimated threshold, $\bar{P}_{th}$, and the false alarm probability, p_f, for each $n = 100, (100), 800$. For each considered p_m, the estimated p_f decreases with n and can be approximated as ab^n, where the parameters a and b were obtained from the p_f estimates for $n = 100, (10), 1000$ and are presented in Table 1.

The parameters a and b were estimated on the first 20 points for $p_m = 0.1$, on the first 20 and 40 points for $p_m = 0.01$, and on the first 20, 40, and 60 points for $p_m = 0.001$, respectively. The most reliable estimates were obtained for the first 20 points in all the cases. To be on the conservative side, the false alarm probabilities are approximated by using the maximum values of b as

$$p_f^{0.1}(n) \approx 0.78 \cdot 0.98^n, \qquad p_f^{0.01}(n) \approx 1.29 \cdot 0.988^n,$$
$$p_f^{0.001}(n) \approx 1.35 \cdot 0.993^n. \tag{33}$$

In view of (33) and (32), the required output string length for a successful correlation attack is then approximately given as

$$n \geq 34.3r_i - 12.3, \tag{34}$$
$$n \geq 57.4r_i - 21.1, \tag{35}$$
$$n \geq 98.7r_i - 42.7, \tag{36}$$

for $p_m = 0.1$, $p_m = 0.01$, and $p_m = 0.001$, respectively.

7.2. *Final Reconstruction*

In the correlation attack, we choose $p_m = 0.1$ and thus obtain multiple candidates for the initial state of LFSR$_i$ which are then ranked in order of decreasing normalized edit probabilities. Candidates for the initial state of the other shift register LFSR$_j$, $j \neq i$, can be produced in the same way by using the edit probability for one input string, regardless of the initial state candidates obtained for LFSR$_i$.

Alternatively, one may use the edit probability for two input strings defined in Section 3.2 (the way the input strings X^{n+2} and Y^{n+2} are assigned to LFSR$_i$ and LFSR$_j$ is not important due to the symmetry). Namely, for each initial state candidate obtained for LFSR$_i$ and each assumed initial state of LFSR$_j$, $j \neq i$, compute the normalized edit probability for two input strings, $\bar{P}(X^{n+2}, Y^{n+2}; Z^n)$, and then apply a similar statistical hypothesis testing procedure as above to obtain the initial state candidates for LFSR$_j$. In this case:

- $\mathbf{H_0}$ (*correlated case*): X^{n+2}, Y^{n+2}, and C^{n+1} are purely random and independent and $Z^n = \dot{F}^n(X^{n+2}, Y^{n+2}, C^{n+1})$.
- $\mathbf{H_1}$ (*independent case*): X^{n+2}, $\hat{X}^{n+2}$, Y^{n+2}, and C^{n+1} are purely random and independent and $Z^n = \dot{F}^n(\hat{X}^{n+2}, Y^{n+2}, C^{n+1})$.

As the edit probability for two input strings makes use of the additional information about one of the input strings, the required output string length for the same number of multiple candidates is in this case smaller than for the edit probability for one input string. Note that the search through the initial states of LFSR$_j$ is reduced if a prefix of length smaller than r_j of the considered input string is found such that the resulting edit probability is equal to zero. In addition, this method also reduces the number of candidate initial states for the first shift register LFSR$_i$.

Accordingly, we obtain a relatively small number of candidate initial state pairs for LFSR$_1$ and LFSR$_2$ in time $O(2^{\max(r_1,r_2)+2\log_2 \max(r_1,r_2)})$.

Both component candidate initial states are ranked in order of decreasing normalized edit probabilities. The (unique) correct pair along with the correct initial state of LFSR$_3$ can then all be reconstructed by the edit distance method from [6]. For each candidate pair, by backtracking through the matrix of the corresponding partial edit distances one can recover all possible clock-control strings C^n that together with the corresponding input strings result in a given output string. Note that such clock-control strings need not exist (nonzero edit distance) if a candidate pair is obtained by using the edit probability for one input string for both LFSRs. The average number of such clock-control strings of length n per candidate pair can be approximated as $m_n \approx 1.2 \cdot 2^{0.27n}$ (see [6]). Alternatively, if the candidate initial states for the other shift register LFSR$_j$ are obtained by the edit probability for two input strings, all possible clock-control strings for each candidate initial state pair for LFSR$_1$ and LFSR$_2$ can also be obtained by backtracking through the matrix of positive partial edit probabilities.

So, pick $n = r_3$ and then test each obtained initial state triple for all the LFSRs by generating the corresponding ASG output sequence and by comparing it with the given one. In this final stage, the unique solution for the initial states of all the LFSRs is thus found in $O(2^{0.27r_3})$ time. This is considerably smaller than $O(2^{r_3})$ which corresponds to exhaustive search over the initial states of LFSR$_3$.

7.3. *Experimental Correlation Attacks*

A number of computer simulations were conducted to show that the above correlation attack can work in practice.[3] Only primitive feedback polynomials were used for all the LFSRs. The correlation attack was performed by using the edit probability for one input string to recover the initial state candidates for both LFSR$_1$ and LFSR$_2$.

[3] For the experiments, we used a Macintosh PowerPC 8600 with a PowerPC 604 processor, clock frequency 180 MHz, and 256 MB of RAM.

Table 2. ASG: Experimental results.

	$n = 200$	$n = 400$	$n = 600$	(r_1, r_2, r_3)
N_1, N_2	1380, 463	106, 37	22, 25	
$N_{1,2}, N_3$	6, 40	6, 40	6, 40	(15, 14, 20)
k_1, k_2	43, 7	1, 2	1, 2	
N_1, N_2	1125, 1168	36, 33	28, 31	
$N_{1,2}, N_3$	3, 8100	3, 8100	3, 8100	(15, 16, 24)
k_1, k_2	1, 1	1, 1	1, 1	
N_1, N_2	479, 289	28, 33	23, 23	
$N_{1,2}, N_3$	4, 3168	4, 3168	4, 3168	(16, 16, 28)
k_1, k_2	7, 1	6, 1	1, 1	
N_1, N_2	4353, 2243	125, 46	17, 20	
$N_{1,2}, N_3$	2, 2484	2, 2484	2, 2484	(18, 17, 30)
k_1, k_2	3, 12	1, 37	1, 2	

Some examples of the experimental results obtained are shown in Table 2. In each experiment, described by the shift register lengths (r_1, r_2, r_3), for any chosen n, N_i denotes the number of candidate initial states for LFSR$_i$, $i = 1, 2$, $N_{1,2}$ denotes the number of candidate initial state pairs (among $N_1 N_2$ of them) that passed the zero edit distance test, k_i denotes the rank of the normalized edit probability corresponding to the correct initial state of LFSR$_i$, $i = 1, 2$, and N_3 stands for the number of clock-control strings of length r_3 that had to be tested per correct initial state pair. In each experiment, a unique solution for the LFSR initial states was obtained. Notice that although a relatively small number of multiple candidates for the initial states of LSFR$_1$ and LFSR$_2$ did appear, the correct initial states always ranked the best or very close to the best provided n was sufficiently large. As indicated by (34), it was observed that N_i was approximately minimized by using $n \approx 40r_i$, $i = 1, 2$. However, in practice, it appears that k_i reduces to one or to a very small integer even if $n \approx 20r_i$.

8. Correlation Attack on BSGG

The general lines of the correlation attack are described in Section 6. Here we specify the details adapted to the BSGG and provide experimental results obtained by computer simulations.

The correlation attack consists of two phases. The goal of the first phase is to recover the initial state of LFSR$_1$ by using the (normalized) edit probability introduced in Section 4. A number of candidates for the initial state of LFSR$_1$ are obtained in this way. Then, in the second phase, the correct initial states of LFSR$_1$ and LFSR$_2$ are both reconstructed. In view of the state diagram of the BSGG, one may expect that there are at most two solutions for the LFSR initial states and that the unique solution is more likely. Note that the correlation attack on LFSR$_2$ based on the corresponding edit probability is expected to be equally efficient because of equal correlation coefficients and equal LFSR lengths.

8.1. *Statistical Discrimination and Output String Length*

The normalized edit probability $\bar{P}(X^{n+L'}; Z^n)$ is computed for the output string Z^n and an input string $X^{n+L'}$, where Z^n is constituted by the first n bits of the first derivative of the known keystream sequence and $X^{n+L'}$ is the first $n + L'$ output bits generated by the LFSR$_1$ recursion from an assumed initial state. The probability distribution of $\bar{P}(X^{n+L'}; Z^n)$ is considered under the following two hypotheses:

- $\mathbf{H_0}$ (*correlated case*): $X^{n+L'}$ and $Y^{n+L'}$ are purely random and independent and $Z^n = \dot{\mathcal{G}}^n(X^{n+L'}, Y^{n+L'})$.
- $\mathbf{H_1}$ (*independent case*): $X^{n+L'}$ and Z^n are purely random and independent.

We conducted systematic experiments and produced histograms of the two distributions for each $n = 100, (10), 800$ on random samples generated from 1000 pairs $(X^{n+L'}, Z^n)$ according to $\mathbf{H_0}$ (without essential difference, we chose $L = 100$) and $\mathbf{H_1}$, respectively. They show that the separation between the distributions increases with n. It thus turns out that the normalized edit probability is much larger under $\mathbf{H_0}$ than under $\mathbf{H_1}$, for a sufficiently large n. For illustration, Tables 1B and 2B given in Appendix B display the observed minimum, maximum, mean, and median values along with the standard deviation of the normalized edit probability, for each $n = 100, (100), 800$, under $\mathbf{H_1}$ and $\mathbf{H_0}$, respectively.

For $p_\mathrm{m} = 0.1$ and $p_\mathrm{m} = 0.05$, Table 3B given in Appendix B displays the estimated threshold, $\bar{P}_\mathrm{th}$, and false alarm probability, p_f, for each $n = 100, (100), 800$. For each considered p_m, the estimated p_f decreases with n and can be approximated as ab^n, where the parameters a and b were obtained from the p_f estimates for $n = 100, (10), 1000$ and are presented in Table 3.

The parameters a and b were estimated on the first 10, 20, and 25 points for both $p_\mathrm{m} = 0.1$ and $p_\mathrm{m} = 0.05$, respectively. The most reliable estimates were obtained for the first 10 points. To be on the conservative side, the false alarm probabilities are approximated by using the maximum values of b as

$$p_\mathrm{f}^{0.1}(n) \approx 0.542 \cdot 0.986^n, \qquad p_\mathrm{f}^{0.05}(n) \approx 0.520 \cdot 0.990^n. \tag{37}$$

In addition, we also conducted experiments in which the correlated samples were generated according to the approximative model for the BSGG used for defining the edit probability. Namely, $\mathbf{H_0}$ was replaced by $\mathbf{H_0'}$ where $X^{n+L'}$, Y^{n+2}, and R^{2n+2} are purely random and independent and $Z^n = \dot{G}^n(X^{n+L'}, Y^{n+2}, R^{2n+2})$. Interestingly, it turns out that the statistical discrimination between $\mathbf{H_1}$ and $\mathbf{H_0}$ is larger than the one between $\mathbf{H_1}$ and $\mathbf{H_0'}$. This can be regarded as a consequence of the randomization introduced by the auxiliary random clock-control string, despite the fact that $\mathbf{H_0'}$ is more suited to the edit probability used.

Table 3. BSGG: Estimation of a and b on 10*, 20**, and 25*** points.

p_m	a^*	a^{**}	a^{***}	b^*	b^{**}	b^{***}
0.1	0.542	0.585	0.551	0.986	0.986	0.986
0.05	0.520	0.736	0.731	0.990	0.987	0.987

In view of (37) and (32), the required output string length for a successful correlation attack is then approximately given as

$$n \geq 49.2L - 43.4, \tag{38}$$

$$n \geq 69.0L - 65.1, \tag{39}$$

for $p_m = 0.1$ and $p_m = 0.05$, respectively.

8.2. *Final Reconstruction*

In the first phase of the correlation attack, a relatively small number of candidate initial states for $LFSR_1$ are obtained in time $O(2^{L+3\log_2 L})$. They are then ranked and tested in order of decreasing normalized edit probabilities. Namely, each candidate initial state for $LFSR_1$ is associated with each of the $2^L - 1$ possible initial states of $LFSR_2$ and the corresponding output sequence is compared with the given BSGG output sequence. In this phase the correct LFSR initial states are thus found in time $O(2^L)$.

8.3. *Experimental Correlation Attacks*

The correlation attack was tested on short LFSRs by computer simulations, which verified that the attack can work in practice. Some results of our experiments are shown in Table 4. For $p_m = 0.1$, the required output string length was first estimated by (38) for $L = 14$, 15, and 16. The experiments were also repeated by halving this string length. The good results obtained for $L = 14$, 15, and 16 when reducing n to $n/2$ (and time reduction) motivated the choice of $n = 500$ (400), instead of $n = 800$ (400), for $L = 17$. The thresholds for the normalized edit probability were obtained from the data collected for $n = 100$, (10), 800 (see Table 3B for $n = 100$, (100), 800). For $n = 325$, 375, we used interpolation.

We counted the number of $LFSR_1$ states giving rise to a normalized edit probability not smaller than the given threshold (Candidates). For every candidate initial state for $LFSR_1$ we searched for a companion initial state of $LFSR_2$ and counted the joint solutions (Solutions). Table 4 shows that a unique joint solution was always found. Finally, we determined the position of the $LFSR_1$ component of this solution in the list of the initial state candidates for $LFSR_1$ ranked in order of decreasing normalized edit probabilities (Rank).

Table 4. BSGG: Experimental results.

L	n	Threshold	Candidates	Solutions	Rank
14	650 (325)	11,290 (75)	20 (51)	1 (1)	1 (1)
15	700 (350)	21,720 (109)	8 (72)	1 (1)	1 (5)
16	750 (375)	46,070 (145)	30 (171)	1 (1)	1 (1)
17	500 (400)	741 (180)	35 (139)	1 (1)	26 (29)

As for the initial state candidates for LFSR_1, we found that a candidate is very likely obtainable from the correct LFSR_1 initial state by a small positive or negative phase shift.

9. Correlation Attack on Alleged A5

The general lines of the correlation attack are described in Section 6. Here we specify the details adapted to the alleged A5 generator and discuss the possibility of obtaining the experimental results by computer simulations.

The correlation attack consists of two phases. The goal of the first phase is to recover the initial states of the shortest two LFSRs, LFSR_1 and LFSR_2, by using the (normalized) edit probability introduced in Section 5. It is assumed that the LFSRs are indexed in order of increasing lengths. A number of candidates for the initial states of LFSR_1 and LFSR_2 are obtained in this way. Then, in the second phase, the correct initial states of all three LFSRs are reconstructed. In view of the fact that the next-state function of the alleged A5 is not one-to-one, it is argued in [3] and [5] that several different LFSR initial state triples may give rise to the same output sequence, so that the solution for the LFSR initial states may not be unique. In any case, the number of (equivalent) solutions is small and all of them can be easily obtained from any one of them by the branching method [3], [5].

9.1. *Reducing Space and Time Complexities*

In order to avoid slowing down the computations, the operating memory of a computer system should be used to store the partial edit probabilities. In this regard, the space complexity $O(n^3)$ is prohibitively high if the string length n is of the order of thousands, as could be expected to be needed in a realistic correlation attack. We now propose a method to reduce the space complexity for the edit probability computation to $O(n^{3/2})$.

As the 4-valued clock-control string is purely random if the input strings are independent and purely random, $w_{ij}(C^s)$ is then a binomially distributed random variable with the expected value $\mu(s) = s/4$ and the standard deviation $\sigma(s) = \sqrt{3s}/4$. Accordingly, for most input strings $X^{n+\tau_1'}$ and $Y^{n+\tau_2'}$, the most significant values of the partial edit probability $P(l_1, l_2, l_3, s)$ are concentrated around the point $(l_1, l_2, l_3) \approx ((s+1)/4, (s+1)/4, (s+1)/4)$, while the others are considerably smaller. So, the idea is to compute only the significant values while the others are set to zero.

More precisely, $P(l_1, l_2, l_3, s)$ is computed only for $(l_1, l_2, l_3) \in \mathcal{L}_s(M)$ defined as follows. First compute $r_1(n) = \lceil \mu(n+1) - 3\sigma(n+1) \rceil$ and $M = \lceil 6\sigma(n+1) \rceil$ and set $r_2(n) = r_1(n) + M$ (for large n, $[r_1(n), r_2(n)] \subseteq [0, n+1]$). Second, for any $1 \le s < n$, compute $r_1(s)$ and $r_2(s)$ as follows. If $s+1 \le M$, then $r_1(s) = 0$ and $r_2(s) = s+1$. If $M < s+1 \le 2M$, then $r_1(s) = 0$ and $r_2(s) = M$. If $s+1 > 2M$, then $r_1(s) = \lceil (s+1)/4 - M/2 \rceil$ and $r_2(s) = r_1(s) + M$. Accordingly, for any $1 \le s \le n$, $\mathcal{L}_s(M)$ is defined as the set of all (l_1, l_2, l_3) such that $r_1(s) \le l_1, l_2, l_3 \le r_2(s)$ and $l_1 + l_2 + l_3 \le s+1$.

The same recursion (30) is used for the computation, with a difference that the involved partial probability values for the preceding value of s that have not been computed are set to zero. The space complexity is then $O(M^3) = O(n^{3/2})$, and the time complexity is reduced to $O(n^{5/2})$.

9.2. *Statistical Discrimination and Output String Length*

The normalized edit probability $\bar{P}(X^{n+\tau_1'}, Y^{n+\tau_2'}; Z^n)$ is computed for the output string Z^n and input strings $X^{n+\tau_1'}$ and $Y^{n+\tau_2'}$, where Z^n is constituted by the first n bits of the first derivative of the known keystream sequence and $X^{n+\tau_1'}$ and $Y^{n+\tau_2'}$ are the first $n+\tau_1'$ and $n+\tau_2'$ output bits generated by the LFSR$_1$ and LFSR$_2$ recursions from assumed initial states, respectively. The edit probability can be modified according to the space reduction method.

The probability distribution of $\bar{P}(X^{n+\tau_1'}, Y^{n+\tau_2'}; Z^n)$ is considered under the following two hypotheses:

- $\mathbf{H_0}$ (*correlated case*): $X^{n+\tau_1'}$, $Y^{n+\tau_2'}$, and $U^{n+\tau_3'}$ are purely random and independent and $Z^n = \dot{\mathcal{G}}^n(X^{n+\tau_1'}, Y^{n+\tau_2'}, U^{n+\tau_3'})$.
- $\mathbf{H_1}$ (*independent case*): $X^{n+\tau_1'}$, $Y^{n+\tau_2'}$, and Z^n are purely random and independent.

Alternatively, the correlated case can also be modeled by $\mathbf{H_0'}$ where $X^{n+\tau_1'}$, $Y^{n+\tau_2'}$, U^{n+2}, and R^{n+1} are purely random and independent and $Z^n = \dot{G}^n(X^{n+\tau_1'}, Y^{n+\tau_2'}, U^{n+2}, R^{n+1})$.

According to the available computational resources, we conducted systematic experiments for the modified edit probability and produced histograms of the two distributions for each $n = 250, (250), 2000$ on random samples generated from 100 random triples $(X^{n+\tau_1'}, Y^{n+\tau_2'}, Z^n)$ according to $\mathbf{H_0'}$ (without essential difference, we chose τ_i' to be around 15) and $\mathbf{H_1}$, respectively. They show that the separation between the distributions slowly increases with n. For $p_{\mathrm{m}} = 0.1$ and each n, we computed the threshold and the false alarm probability, p_{f}.

In view of the experimental results for the BSGG, we anticipate that for large n the false alarm probability also exponentially decreases as ab^n, $b < 1$. However, in the range of n considered, p_{f} very slowly decreases with n, so that the estimates of p_{f} obtained did not enable us to produce reliable estimates of the parameters a and b. This means that larger values of n have to be explored and that b is much closer to 1 than in the case of the BSGG. According to (32), the required output string length for a successful correlation attack is linear in $r_1 + r_2$, but the multiplicative constant is fairly large.

9.3. *Final Reconstruction*

In the first phase of the correlation attack, we obtain a relatively small number of candidate initial state pairs for LFSR$_1$ and LFSR$_2$ in time $O(2^{r_1+r_2+2.5\log_2(r_1+r_2)})$, where the multiplicative constant is expected to be fairly large. They are then ranked and tested in order of decreasing normalized edit probabilities. Namely, for each candidate initial state pair, all possible initial states of LFSR$_3$ are tested by generating the output sequence and comparing it with the known keystream sequence. In this phase, all the solutions for the correct LFSR initial states are thus found in time $O(2^{r_3})$.

If $r_3 > r_1 + r_2$, then the time complexity of the final phase may be reduced by a more sophisticated method based on an appropriate edit distance. This edit distance can be defined on the basis of the same edit transformation as the edit probability for one independent binary clock-control string (modified to reduce the space complexity). For each candidate initial state pair, compute this edit distance and discard the pair if it differs from zero. (The zero value indicates that the pair is consistent with the given output string in the underlying model.) Then recover all the strings U^{m+2} and R^{m+1} that are, together with the candidate input strings, consistent with the given output string of length m, by backtracking through the array of the corresponding partial edit distances following the minimum edit distance paths when decreasing s. By employing the fact that R^{m+1} can be derived from $U_{\tau_3}^{m+\tau_3}$ by stop/go clocking, one can thus reconstruct the segments composed of the last r_3 effectively used bits of all the consistent strings $U^{m+\tau_3'}$ ($m \geq 4r_3/3$). Since the number of such segments is (much) smaller than 2^{r_3}, exhaustive search over all the initial states of LFSR$_3$ is thus avoided.

10. Conclusions

It is pointed out that the stop/go clocking in certain keystream generators based on stop/go clocked LFSRs can be viewed as a random edit transformation of a number of input strings into one output string. The keystream generators include the ASG, the BSGG, and the alleged A5. The input strings correspond to the output sequences of a number of LFSRs when regularly clocked and the output string corresponds to the first derivative of the output sequence of the respective generator. The output sequences of the remaining LFSRs are assumed to be purely random and independent. For the BSGG and the alleged A5, some additional purely random strings are also introduced. The related edit probabilities are defined and the recursive algorithms for their efficient computation are derived.

It is shown how the edit probabilities can be used to mount statistically optimal correlation attacks on a number of LFSRs in each of the schemes considered. The correlation attacks require the computation of the corresponding edit probability for all possible initial states of the LFSRs targeted by the attacks. In the final stage, the initial states of all the LFSRs are reconstructed. A statistical hypothesis testing method for estimating the known keystream sequence length needed for a successful correlation attack is developed. The method requires experiments by computer simulations. It turns out that this length is linear in the total length of the targeted LFSRs. A divide-and-conquer effect is achievable if the total length of the remaining LFSRs is not relatively small.

Systematic experiments including successful correlation attacks on relatively short LFSRs are conducted for the ASG and the BSGG. The results demonstrate that in order to prevent the correlation attacks the total length of the targeted LFSRs should be sufficiently long.

In general, the methodology developed in this paper shows that the security against correlation attacks of keystream generators involving clock-controlled shift registers can be analyzed by using special edit probabilities adapted to the structure considered. A challenging research field is investigating the possibility of the corresponding fast correlation attacks which would not require the exhaustive search through the initial states of the targeted LFSRs.

Appendix A. Statistical Data for ASG

Table 1A. Statistics of $\bar{P}$ on 10,000 independent pairs (X^{n+2}, Z^n).

n	Min	Max	Mean	Median	Std dev
100	6.059E-10	7.749E2	1.825E0	5.051E-2	1.406E1
200	5.613E-13	1.207E4	2.493E0	2.646E-3	1.227E2
300	2.514E-14	2.320E3	6.438E-1	1.696E-4	2.832E1
400	4.979E-19	9.943E2	4.923E-1	1.273E-5	1.646E1
500	2.066E-19	4.170E2	2.194E-1	1.121E-6	6.085E0
600	2.761E-20	7.503E1	2.898E-2	8.78E-8	9.7E-1
700	1.535E-23	5.366E1	1.477E-2	8.478E-9	6.352E-1
800	8.513E-24	2.454E1	4.59E-3	6.947E-10	2.645E-1

Table 2A. Statistics of $\bar{P}$ on 10,000 correlated pairs (X^{n+2}, Z^n).

n	Min	Max	Mean	Median	Std dev
100	1.162E-3	1.024E6	9.378E2	4.773E1	1.246E4
200	3.717E-3	1.772E8	1.502E5	7.643E2	2.893E6
300	6.552E-3	1.4E11	3.625E7	1.092E4	1.532E9
400	1.648E-2	1.422E14	1.86E10	1.731E5	1.429E12
500	1.006E-3	7.071E15	1.697E12	2.483E6	9.125E13
600	3.388E-3	4.436E18	4.696E14	3.279E7	4.438E16
700	1.170E-3	2.514E20	2.542E16	5.479E8	2.514E18
800	2.401E-2	7.198E20	1.889E17	7.999E9	8.513E18

Table 3A. Estimation of thresholds and false alarm probabilities.

n	$\bar{P}_{\mathrm{th}}^{0.1}$	$\bar{P}_{\mathrm{th}}^{0.01}$	$\bar{P}_{\mathrm{th}}^{0.001}$	$p_{\mathrm{f}}^{0.1}$	$p_{\mathrm{f}}^{0.01}$	$p_{\mathrm{f}}^{0.001}$
100	2.193E0	1.680E-1	2.005E-2	1.015E-1	3.491E-1	6.127E-1
200	1.038E1	3.098E-1	1.308E-2	1.61E-2	1.127E-1	3.459E-1
300	5.684E1	1.093E0	6.713E-2	1.0E-3	2.29E-2	9.48E-2
400	4.047E2	3.084E0	1.7E-1	3E-4	5.8E-3	2.65E-2
500	2.729E3	9.323E0	1.061E-1	0	2.9E-3	1.95E-2
600	2.295E4	4.723E1	5.289E-1	0	2E-4	4.2E-3
700	2.150E5	3.290E2	1.398E0	0	0	1.2E-3
800	1.19E6	1.009E3	5.684E0	0	0	2E-4

Appendix B. Statistical Data for BSGG

Table 1B. Statistics of $\bar{P}$ on 1000 independent pairs $(X^{n+L'}, Z^n)$.

n	Min	Max	Mean	Median	Std dev
100	6.341E-9	3.339E2	2.314E0	1.43E-1	1.294E1
200	4.307E-15	3.952E2	2.121E0	1.597E-2	1.69E1
300	4.279E-15	2.774E2	1.632E0	2.674E-3	1.347E1
400	2.669E-14	6.531E2	1.2E0	4.36E-4	2.168E1
500	2.246E-14	7.893E1	2.456E-1	1.13E-4	2.82E0
600	6.1E-20	3.019E2	4.643E-1	2.023E-5	9.68E0
700	3.915E-18	1.598E4	1.613E1	3.771E-6	5.055E2
800	5.145E-19	7.301E1	1.669E-1	7.699E-7	2.848E0

Table 2B. Statistics of $\bar{P}$ on 1000 correlated pairs $(X^{n+L'}, Z^n)$.

n	Min	Max	Mean	Median	Std dev
100	7.637E-2	7.097E3	1.097E2	2.486E1	4.006E2
200	8.051E-2	1.703E6	6.538E3	2.033E2	7.162E4
300	5.31E-2	3.59E7	1.269E5	1.863E3	1.62E6
400	3E-1	2.658E8	1.759E6	1.056E4	1.438E7
500	3.053E0	7.872E10	1.213E8	1.108E5	2.573E9
600	8.349E-1	3.3E12	6.27E9	6.357E5	1.222E11
700	4.942E1	2.123E12	4.933E9	5.135E6	7.462E10
800	3.484E0	1.987E13	7.507E10	2.374E7	8.822E11

Table 3B. Estimation of thresholds and false alarm probabilities.

n	$\bar{P}_{\mathrm{th}}^{0.1}$	$\bar{P}_{\mathrm{th}}^{0.05}$	$p_{\mathrm{f}}^{0.1}$	$p_{\mathrm{f}}^{0.05}$
100	3.122E0	1.893E0	1.22E-1	1.67E-1
200	1.214E1	5.655E0	2.7E-2	4.8E-2
300	4.138E1	1.687E1	1.1E-2	1.9E-2
400	1.808E2	5.709E1	1E-3	3E-3
500	7.413E2	2.822E2	0	0
600	7.164E3	1.505E3	0	0
700	2.172E4	3.835E3	0	1E-3
800	7.587E4	1.891E4	0	0

References

[1] A. Biryukov, A. Shamir, and D. Wagner, Real time cryptanalysis of A5/1 on a PC, *Fast Software Encryption - FSE*, 2000, Lecture Notes in Computer Science, vol. 1978, B. Schneier, ed., Springer-Verlag, Berlin, pp. 1–18, 2001.

[2] W. G. Chambers, On random mappings and random permutations, *Fast Software Encryption - FSE* '94, Lecture Notes in Computer Science, vol. 1008, B. Preneel, ed., Springer-Verlag, Berlin, pp. 22–28, 1995.

[3] J. Dj. Golić, Cryptanalysis of alleged A5 stream cipher, *Advances in Cryptology - EUROCRYPT* '97, Lecture Notes in Computer Science, vol. 1233, W. Fumy, ed., Springer-Verlag, Berlin, pp. 239–255, 1997.

[4] J. Dj. Golić, Recent advances in stream cipher cryptanalysis, *Publications de l'Institut Mathematique*, vol. 64/78, pp. 183–204, 1998.

[5] J. Dj. Golić, Cryptanalysis of three mutually clock-controlled stop/go shift registers, *IEEE Transactions on Information Theory*, vol. 46, pp. 1081–1090, May 2000.

[6] J. Dj. Golić and R. Menicocci, Edit distance correlation attack on the alternating step generator, *Advances in Cryptology - CRYPTO* '97, Lecture Notes in Computer Science, vol. 1294, B. Kaliski, ed., Springer-Verlag, Berlin, pp. 499–512, 1997.

[7] J. Dj. Golić and M. Mihaljević, A generalized correlation attack on a class of stream ciphers based on the Levenshtein distance, *Journal of Cryptology*, vol. 3(3), pp. 201–212, 1991.

[8] J. Dj. Golić and L. O'Connor, Embedding and probabilistic correlation attacks on clock-controlled shift registers, *Advances in Cryptology - EUROCRYPT* '94, Lecture Notes in Computer Science, vol. 950, A. De Santis, ed., Springer-Verlag, Berlin, pp. 230–243, 1995.

[9] J. Dj. Golić and S. Petrović, A generalized correlation attack with a probabilistic constrained edit distance, *Advances in Cryptology - EUROCRYPT* '92, Lecture Notes in Computer Science, vol. 658, R. A. Rueppel, ed., Springer-Verlag, Berlin, pp. 472–476, 1993.

[10] D. Gollmann and W. G. Chambers, Clock-controlled shift registers: a review, *IEEE Journal on Selected Areas in Communications*, vol. 7, pp. 525–533, May 1989.

[11] C. G. Günther, Alternating step generators controlled by de Bruijn sequences, *Advances in Cryptology - EUROCRYPT* '87, Lecture Notes in Computer Science, vol. 304, D. Chaum and W. L. Price, eds., Springer-Verlag, Berlin, pp. 5–14, 1988.

[12] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, FL, 1997.

[13] B. Schneier, *Applied Cryptography*, Wiley, New York, 1996.

[14] K. Zeng, C. H. Yang, and T. R. N. Rao, Large primes in stream-cipher cryptography, *Advances in Cryptology - AUSCRYPT* '90, Lecture Notes in Computer Science, vol. 453, J. Seberry and J. Pieprzyk, eds., Springer-Verlag, Berlin, pp. 194–205, 1990.

[15] K. Zeng, C. H. Yang, D. Y. Wey, and T. R. N. Rao, Pseudorandom bit generators in stream-cipher cryptography, *IEEE Computer*, vol. 24(2), pp. 8–17, Feb. 1991.