A Low-Complexity and High-Performance Algorithm for the Fast Correlation Attack*

Miodrag J. Mihaljević[1], Marc P.C. Fossorier[2], and Hideki Imai[3]

[1] Mathematical Institute, Serbian Academy of Science and Arts,
Kneza Mihaila 35, 11001 Belgrade, Yugoslavia
email: `miodragm@turing.mi.sanu.ac.yu`

[2] Department of Electrical Engineering, University of Hawaii,
2540 Dole St., Holmes Hall 483, Honolulu, HI 96822, USA
email: `marc@spectra.eng.hawaii.edu`

[3] University of Tokyo, Institute of Industrial Science,
7-22-1, Roppongi, Minato-ku, Tokyo 106-8558, Japan
email: `imai@iis.u-tokyo.ac.jp`

Abstract. An algorithm for cryptanalysis of certain keystream generators is proposed. The developed algorithm has the following two advantages over other reported ones: (i) it is more powerful and (ii) it provides a high-speed software implementation, as well as a simple hardware one, suitable for high parallel architectures. The novel algorithm is a method for the fast correlation attack with significantly better performance than other reported methods, assuming a lower complexity and the same inputs. The algorithm is based on decoding procedures of the corresponding binary block code with novel constructions of the parity-checks, and the following two decoding approaches are employed: the a posterior probability based threshold decoding and the belief propagation based bit-flipping iterative decoding. These decoding procedures offer good trade-offs between the required sample length, overall complexity and performance. The novel algorithm is compared with recently proposed improved fast correlation attacks based on convolutional codes and turbo decoding. The underlying principles, performance and complexity are compared, and the gain obtained with the novel approach is pointed out.

Keywords: stream ciphers, keystream generators, linear feedback shift registers, fast correlation attack, decoding.

1 Introduction

An important method for attack or security examination of certain stream ciphers based on nonlinear combination keystream generators composed of several linear feedback shift registers (LFSR's) (see [11], for example) are: basic correlation attack [18], and particularly the fast correlation attacks considered in a

* This work was supported by JSPS Grant RFTF 96P00604 and NSF Grant CCR-97-32959

number of papers, including [12], [20], [13], [14], [2], [3], [8], [9] and [6]. Developing or improving techniques for realization of the fast correlation attack is a standard cryptologycal problem.

The basic ideas of all reported fast correlation attacks include the following two main steps:
- Transform the cryptographic problem into a suitable decoding one;
- Apply (devise) an appropriate decoding algorithm.

There are two main approaches for realization of the fast correlation attack. The first one is based on decoding techniques for block codes (introduced in [12] and [19]), and the second one is based on decoding techniques for convolutional codes (recently proposed in [8] and [9]).

The main underlying ideas for the fast correlation attacks based on linear binary block codes decoding is the iterative decoding principle introduced in [4]. For example, all the fast correlation attacks reported in [12], [19], [13], [20], [14], [2], and [3], could be considered as variants of iterative decoding based on simple bit-flipping (BF) [4] or iterative extensions of *a posterior probability* (APP) decoding [10]. Most of these methods (practically all except the method from [13]) are restricted on the LFSR feedback polynomials of low weight. Due to the established advantages of belief propagation (BP) based iterative decoding over iterative APP (see [5], for example), the application of BP based iterative decoding for realization of the fast correlation attack has been recently reported in [6]. The main goal of [6] was to report the potential gain and its origins when BP based iterative decoding is employed instead of APP based decoding, assuming the same construction method of the parity-checks and the same overall structure of the algorithm for fast correlation attack. A comparison of the iterative decoding approaches based on simple, APP and BP based decodings for the fast correlation attack is reported in [15].

New methods for fast correlation attack based on the theory of convolutional codes are given in [8]-[9]. They can be applied to arbitrary LFSR feedback polynomials, in opposite to the previous methods, which mainly focus on feedback polynomials of low weight. The proposed algorithm transforms a part of the code **C** steaming from the LFSR sequence into a convolutional code, based on finding suitable parity check equations for **C**. The approach considers a decoding algorithm that includes memory, but still has a low decoding complexity. With respect to the previous methods, this allows looser restrictions on the parity check equations that can be used, leading to many more equations. As the final decoding method, the Viterbi algorithm with memory orders of 10-15 was used. The results reported in [8] improve significantly the few previous results for high weight feedback polynomials, and are in many cases comparable with that corresponding to low weight feedback polynomials. Further developments of the idea for fast correlation attack based on decoding of certain convolutional codes are presented in [9] where new methods employing the techniques used for constructing and decoding turbo codes are proposed. The most powerful technique presented in [9] is based on the turbo decoding approach with M component convolutional codes and iterative APP decoding employing the BCJR algorithm [1] (see also [7]).

Recent interests and the advances in developing algorithms for the fast correlation attack have raised a natural question of further improvements of the fast correlation attack, especially in the light of fast implementations.

The main goal of this paper is to propose an algorithm for the fast correlation attack suitable for a high-speed software implementation, as well as for a simple hardware one. Most existing algorithms can be considered as inappropriate ones for this goal assuming an LFSR feedback polynomial of arbitrary weight. Accordingly, our intention was to develop an algorithm which employs *mod*2 additions and simple logical operations for processing, so that it is suitable for highly parallel architectures and high speed software or hardware implementations. Also, our goal is to propose an algorithm which yields possibility for trade-offs between length of the required sample, overall complexity and performance.

In this paper, a more powerful algorithm for the fast correlation attack with significantly better performance (which does not depend on the LFSR feedback polynomial), assuming the same inputs and lower complexity than other reported methods, is proposed. The proposed algorithm is based on a novel method for constructing the parity-checks, motivated by the approach of [8] and [9], and two decoding approaches of the corresponding binary block code, APP threshold decoding [10] and iterative decoding employing BP-like BF (see [4]). The construction of the parity-checks is based on searching for certain parity-check equations and theirs linear combinations employing the finite-state machine model of an LFSR with primitive characteristic polynomial. The expected numbers of parity-checks per parity bit are derived, showing that a large number of appropriate parity-checks can be constructed. An analysis of the algorithm performance and complexity is presented. The novel algorithm is compared with recently proposed improved fast correlation attacks based on convolutional codes and turbo decoding. The underlying principles, performances and complexities are compared, and the gains obtained with the novel approach are pointed out. It is shown that assuming the same input, the novel algorithm yields better performance and lower complexity than the best algorithm reported up-to-now.

The paper is organized as follows. Section 2 presents preliminaries. Section 3 points out the main underlying results for the construction of a novel algorithm for the fast correlation attack. Complete specification of the proposed algorithm is given in Section 4. Experimental analysis of the performance is presented in Section 5, as well as a discussion of the complexity issue. Comparisons between the recently reported improved fast correlation attacks, and the proposed algorithm are given in Section 6. Finally, the results of this paper are summarized in Section 7.

2 Decoding Concept for the Fast Correlation Attack

Recall that, the correlation means that the mod 2 sum of corresponding outputs of the LFSR and the generator can be considered as a realization of a binary random variable which takes value 0 and 1 with the probabilities $1-p$ and p, respectively, $p \neq 0.5$.

The fast correlation attack on a particular LFSR, with primitive feedback polynomial, in a nonlinear combining generator given the segment of the generator output can be considered as follows:

- The n-bit segment of the output sequence from the length-k LSFR is a codeword of an (n, k) punctured simplex code;
- The corresponding n-bit segment of the nonlinear combination generator output is the corresponding noisy codeword obtained through a BSC with crossover probability p;
- The problem of the LFSR initial state reconstruction, assuming known characteristic polynomial, is equivalent to the problem of decoding after transmission over a BSC with crossover probability p.

The decoding approach employed in this paper is based on combination of a restricted exhaustive search over a set of hypotheses and a one-step or an iterative decoding technique. The exhaustive search is employed in order to provide a possibility for construction of suitable parity-check equations relevant for high performance of complete decoding. This approach could be considered as a particular combination of the minimum distance decoding and another decoding technique.

Recall that a parity-check equation which involves a smaller number of bits is more powerful than a higher weight one. Also note that performance associated with a set of the parity-checks depends on its cardinality as well as on the parity-check weight distribution. Finally, the overall complexity of a decoding procedure depends on the number and weights of the employed parity-checks. Accordingly, from performance and complexity point of views, a favorable situation corresponds to the availability of a large number of low-weight parity-checks.

In the following, x_n, $n = 1, 2, ..., N$, denotes an LFSR output sequence which is a codeword $\mathbf{x}$ of a binary (N, L) punctured simplex code $\mathbf{C}$ where N is codeword length and L is number of information bits. $\mathbf{x}_0 = [x_1, x_2, ..., x_L]$ is the vector of information bits identical to the LFSR initial state; $\{z_n\}$ denotes the degraded sequence $\{x_n\}$ after transmission over a BSC with crossover probability p. Accordingly, $z_n = x_n \oplus e_n \quad , \quad n = 1, 2, ..., N \quad$, where the effect of the BSC with error probability p is modeled by an N-dimensional binary random variable $\mathbf{E}$ defined over $\{0, 1\}^N$ with independent coordinates E_n such that $\Pr(E_n = 1) = p$, $n = 1, 2, \ldots, N$, and e_n is a realization of E_n. Applying a codeword $\mathbf{x} = [x_n]_{n=1}^N \in \mathbf{C}$, to the input of the BSC, we obtain the random variable $\mathbf{Z} = \mathbf{E} \oplus \mathbf{x}$ as a received codeword at its output. Let $\mathbf{z} = [z_n]_{n=1}^N$ and $\mathbf{e} = [e_n]_{n=1}^N$ denote particular values of the random vector variables $\mathbf{Z}$ and $\mathbf{E}$, respectively.

3 Novel Appropriate Parity-Check Sets

This section points out novel sets of the parity-check equations relevant for construction of an algorithm for the fast correlation attack which will be proposed in the next section. Also, this section points out the expected cardinalities of these sets.

3.1 Preliminaries

An LFSR can be considered as a linear finite state machine. Recall that a linear finite state machine is a realization or an implementation of certain linear operator. Accordingly, a state of a length-L LFSR after t clocks is given by the following matrix-vector product over GF(2):

$$\mathbf{x}_t = \mathbf{A}^t \mathbf{x}_0 \ , \quad t = 1, 2, ... \ ,$$

where $\mathbf{x}_t$ is an L dimensional binary vector representing the LFSR state after t clocks, $\mathbf{x}_0$ is an L dimensional binary vector representing the initial LFSR state (in notation that it has index L at the top and index 1 at the bottom), and $\mathbf{A}^t$ is the t-th power over GF(2) of the state transition $L \times L$ binary matrix $\mathbf{A}$. Assuming the LFSR characteristic polynomial $f(u) = 1 + \sum_{i=1}^{L} b_i u^i$, the matrix $\mathbf{A}$ is given by:

$$\mathbf{A} = \begin{bmatrix} b_1 & b_2 & b_3 & ... & & b_L \\ 1 & 0 & 0 & ... & & 0 \\ 0 & 1 & 0 & ... & & . \\ . & . & . & ... & & . \\ 0 & & & ... & 1 & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \mathbf{A}_3 \\ . \\ \mathbf{A}_L \end{bmatrix} , \tag{1}$$

where each $\mathbf{A}_i$, $i = 1, 2, ..., L$, represents a $1 \times L$ binary matrix (a row-vector).

Powers of the matrix $\mathbf{A}$ determine algebraic replica of the LFSR initial state bits, i.e. linear equations satisfied by the bits of the codewords from the dual code. Accordingly, they directly specify the parity-checks.

Since our approach assumes an exhaustive search, over the first B information bits, the parity checks are obtained:
- directly from the powers of the matrix $\mathbf{A}$ corresponding to an arbitrary subset of the first B bits of the LFSR initial state and no more than three bits from the remaining $L - B$ bits of the initial state and the bit of the LFSR output sequence;
- as the *mod*2 sum of any two parity checks determined from the powers of the matrix $\mathbf{A}$ when this sum includes an arbitrary number of the first B bits of the LFSR initial state, at most one bit from the remaining $L - B$ bits of the initial state, and the two bits of the LFSR output sequence.
- as the *mod*2 sum of any three parity checks determined by the powers of matrix $\mathbf{A}$ when this sum includes an arbitrary number of the first B bits of the LFSR initial state, no bit from the remaining $L - B$ bits of the initial state, and the three corresponding bits of the LFSR output sequence.

As pointed out in Section 2, a desirable situation is that corresponding to as many low-weight parity-checks as possible. Following this fact and due to the comparison purposes with recently reported improved fast correlation attacks [8] - [9], we focus our intention mainly to parity-checks of effective weight three (i.e. without considering the first B bits), but also employ some of parity-checks of effective weight four as well. Note finally that parity-checks of an arbitrary weight could be considered as a development of certain reported results, for example [13] and [16].

3.2 Methods for Construction and Specification of the Parity-Check Sets

This section presents two methods for obtaining appropriate sets of parity-checks. The developed methods are related to the *information bits* (Method A) and to the *parity bits* (Method B) of the underlying punctured simplex code.

Method A: Parity-check sets related to the *information bits* of the underlying punctured simplex codeword.

Note that

$$x_{L+n} = \mathbf{A}_1^n \mathbf{x}_0 \ \ , \ \ n = 1, 2, ..., N-L \ \ , \tag{2}$$

where $\mathbf{A}_1^n$ is the first row of the n-th power of the state transition matrix $\mathbf{A}$.

Accordingly, the basic parity-check equations (defined on the noisy sequence) are given by:

$$c_{L+n} = z_{L+n} \oplus \mathbf{A}_1^n \mathbf{z}_0 \ \ , \ \ n = 1, 2, ..., N-L \ \ , \tag{3}$$

where $\mathbf{z}_0 = [z_1, z_2, ..., z_L]$.

Assuming that the first B information bits are known, appropriate parity-check equations for the i-th information bit, $i = B+1, B+2, ..., L$ can constructed according to the following definition.

Definition 1. The set Ω_i of parity-check equations associated with information bit-i is composed of:

– All parity-check equations corresponding to the vectors $\mathbf{A}_1^n$ such that each $\mathbf{A}_1^n$ has arbitrary values in the first B coordinates, has value one at the i-th coordinate, and has two ones in all other information bit coordinates;
– All parity-check equations obtained as the *mod*2 sum of two other basic parity-check equations,

$$(z_m \oplus \mathbf{A}_1^m \mathbf{z}_0) \oplus (z_n \oplus \mathbf{A}_1^n \mathbf{z}_0) \ \ ,$$

where m and n have arbitrary values providing that the vector sum $\mathbf{A}_1^m \oplus \mathbf{A}_1^n$ has arbitrary values in the first B coordinates, value one at the i-th coordinate, and value zero in the all other coordinates.

Note that for given parameters N, L, and B, the sets Ω_i, $i = B+1, B+2, ..., L$, can be constructed in advance through a search procedure in a preprocessing phase, and later used for any particular application with these given parameters.

Method B: Parity-check sets related to the *parity bits* of the underlying punctured simplex codeword.

First, an appropriate form of the parity check matrix of a punctured simplex code is pointed out. Then a method for constructing the parity checks is given and the parity checks to be employed by the algorithm are specified by Definition 2.

Recall, that in Section 2, the fast correlation attack has been modeled by the decoding of an (N, L) punctured simplex code used over a BSC. Accordingly, the following statement points out an appropriate form of the code parity-check matrix. This particular form has a one-to-one correspondence with the finite-state machine model of an LFSR with primitive characteristic polynomial.

Proposition 1. The parity-check matrix $\mathbf{H} = [\, \mathbf{P}^T, \mathbf{I}_{N-L} \,]$ of a punctured simplex code (N, L) with corresponding polynomial $f(u) = 1 + \sum_{i=1}^{L} b_i u^i$, where the binary matrix $\mathbf{P}$ is the $L \times (N - L)$ matrix of parity checks, $\mathbf{P}^T$ is its transpose, and $\mathbf{I}_{N-L}$ is the identity matrix of dimension $(N - L) \times (N - L)$, is specified by the following:

$$\mathbf{P}^T = \begin{bmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \cdot \\ \cdot \\ \mathbf{P}_{N-L} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1^{(1)} \\ \mathbf{A}_1^{(2)} \\ \cdot \\ \cdot \\ \mathbf{A}_1^{(N-L)} \end{bmatrix} , \tag{4}$$

where the m-th row of the matrix $\mathbf{P}^T$, is an L-dimensional row vector $\mathbf{A}_1^{(m)}$ equal to the first row of the m-th power, $\mathbf{A}^m$, of the matrix $\mathbf{A}$ given in (1).

The construction of the parity-checks is based on searching for certain linear combinations of rows in an appropriate form of the parity-check matrix given by Proposition 1. Accordingly, the preprocessing phase of the algorithm includes the construction of the parity-checks according to the following algorithm which generates a set of parity checks for each parity bit. Each parity check includes certain B information bits, and no more than $W + 1$ other arbitrary check bits.

Note that $W + 1$ is used here instead three to illustrate that a straightforward generalization is possible where not only the parity-checks of effective weight equal to three are considered.

Algorithm for the construction of the parity checks

- *Input:* The parity check matrix $\mathbf{H} = [\, \mathbf{P}^T, \mathbf{I}_{N-L} \,]$.
- *Processing Steps:* For each parity bit, generate a set of parity check equations employing the following procedure.
  - For $n = L + 1, L + 2, ..., N$ and each w, $1 \leq w \leq W$, proceed as follows:
    - Calculate the *mod*2-sum of the n-th row of the parity-check matrix $\mathbf{H} = [\mathbf{P}^T, \mathbf{I}_{N-L}]$ and any possible w other rows.

  - If the values at positions $i = B+1, B+2, ..., L$, are all zeros, where $B < L$, is a predetermined parameter, record the considered combination into the set Ω_n^*.
- *Output:* The sets of parity check equations Ω_n^*, $n = L+1, L+2, ..., N$.

Definition 2: The set Ω_n^* generated by the above algorithm is the set of all considered parity-check equations related to the n-th parity bit of codewords in the punctured (N, L) simplex code.

Note that each parity-check in Ω_n^* consists of α of the first B information bits with $0 < \alpha \leq B$, none of the remaining last $L - B$ information bits and at most $W + 1$ of the $N - L$ parity check bits, including bit-n.

3.3 Expected Cardinalities of the Parity-Check Sets

Lemma 1. In any set Ω_i, specified by the Definition 1, $i = B+1, B+2, ..., L$, a tight approximation about the expected number $|\bar{\Omega}|$ of the parity-checks is given by the following:

$$|\bar{\Omega}| = 2^{B-L}[(N-L)\binom{L-B-1}{2} + \binom{N-L}{2}] \ . \tag{5}$$

Note that Lemma 1 motivates the construction of Ω_i given in Definition 1. For each type of check sums in Ω_i, that corresponding to minimum weight with non negligible contribution to $|\bar{\Omega}|$ is chosen.
As an illustration, note that for $N = 40000$, $L = 40$ and $B = 18, 19, 20, 21, 22$, Lemma 1 yields that the expected cardinality, $|\bar{\Omega}|$ is equal to 192, 384, 768, 1534, 3066, respectively.

Lemma 2: In any set Ω_n^*, specified by the Definition 2, $n = L+1, L+2, ..., N$, a tight approximation about the expected number $|\bar{\Omega}^*|$ of the parity-checks is given by the following:

$$|\bar{\Omega}^*| \ = \ 2^{-L+B} \sum_{w=1}^{2} \binom{N-L-1}{w} \ . \tag{6}$$

As an illustration, note that for $L = 40$, and (N, B) = (1024,26), (4096,22), (8192,20), and (16384,18), Lemma 2 yields that the expected cardinalities, $|\bar{\Omega}^*|$ are equal to 29.5, 31.4, 31.7, and 31.9, respectively.

Note that Lemmas 1 and 2 show that Definitions 1 and 2 yield large numbers of the parity-checks relevant for an error-correction procedure.

Also, note that Lemmas 1 and 2 imply that the expected cardinalities of the parity-check sets specified by Definitions 1 and 2 do not depend on the LFSR characteristic polynomial, and particularly on its weight.

4 Novel Algorithm for Fast Correlation Attack

The main underlying principles for construction of the novel fast correlation attack include the following:

- General concepts of linear block codes decoding, and particularly:
  - decoding of information bits only, employing an APP based threshold decoding;
  - iterative decoding of the parity bits employing a reduced complexity BP based iterative decoding.
- A novel method for constructing parity checks of a punctured simplex code based on linear finite state machine model of an LFSR (see [13]);
- The idea (implicitly given in [8]) of employing a partial (restricted) exhaustive search in order to enhance performance of the fast correlation attack. The developed algorithm assumes exhaustive search over the first B information bits in conjunction with appropriate decoding approaches.

According to these principles a novel algorithm for the fast correlation attack (based on a linear block code decoding approach) is proposed. The algorithm is based on the novel methods for constructing the appropriate parity-checks presented in the Section 3, and its processing phase includes the following three techniques: (i) hypothesis testing, (ii) decoding of a punctured simplex code and (iii) correlation check. The algorithm employs two different decoding procedures in order to provide desired trade-offs between necessary length of the sample, i.e. the rate of underlying code, performance and overall complexity.

Algorithm for the Fast Correlation Attack

INPUT:

- values of the parameters N, L, B, and the threshold T;
- the noisy received bits $z_1, z_2, ..., z_N$;
- for each information bit i, $i = B+1, B+2, ..., L$, the set Ω_i of corresponding parity-check equations (constructed in the preprocessing phase based on Definition 1), and for each parity bit n, $n = L+1, L+2, ..., N^*$, $N^* \leq N$, the set Ω_n of corresponding parity-check equations (constructed in the preprocessing phase based on Definition 2).

PROCESSING STEPS:

1. *setting the hypothesis*
   From the set of all possible 2^B binary patterns, select a not previously considered pattern $\hat{x}_1, \hat{x}_2, ..., \hat{x}_B$, for the first B information bits. If no new pattern is available, go to the Output (b).
2. *decoding*
   Employ one of the following two decoding algorithms for estimating a candidate for the information bits (i.e. LFSR initial state):

  - One-Step Decoding Algorithm (OSDA) using parity-checks specified by Definition 1;
  - Iterative Decoding Algorithm (IDA) using parity-checks specified by Definition 2.

3. *correlation check*
   Check if the current estimation of the information bits (obtained from the decoding step) $\hat{\mathbf{x}}_0 = [\hat{x}_1, \hat{x}_2, ..., \hat{x}_L]$, is the true one, according to the following:
   For $\hat{\mathbf{x}}_0$, generate the corresponding sequence $\hat{x}_1, \hat{x}_2, ..., \hat{x}_N$, and calculate $S = \sum_{n=1}^{N} \hat{x}_n \oplus z_n$.
   If $S \leq T$ go to Output (a), otherwise go to Step 1.

OUTPUT:
  (a) the considered vector $\hat{\mathbf{x}}_0$ of information bits is the true one;
  (b) the true vector of information bits is not found.

The threshold scalar T is used for checking a hypothesis over all the information bits. For given N, L, B, p, the threshold T is calculated based on the method presented in [18].

The specifications of the employed decoding algorithms OSDA and IDA are given in the following.

4.1 One-Step Decoding Algorithm - OSDA

OSDA decodes the noisy received sequence $[z_1, z_2, ..., z_N]$ for the (N, L) truncated simplex code employing an APP threshold decoding and the sets Ω_i of parity-check equations, specified by Definition 1, $i = B + 1, B + 2, ..., L$ according to the following.

- *parity-checks calculation*
  For each information bit position i, $i = B + 1, B + 2, ..., L$, calculate the parity-check values employing the parity check equations from the set Ω_i.
- *error-correction*
  For each i, $i = B + 1, B + 2, ..., L$ do the following:
  - if the number of satisfied parity-check equations for the considered information bit is smaller than the threshold $T_1(i)$ set $\hat{x}_i = z_i \oplus 1$, otherwise set $\hat{x}_i = z_i$.

The algorithm employs a vector threshold $\mathbf{T}_1 = [T_1(i)]_{i=B+1}^{L}$ which contains values for the APP threshold decoding of certain information bits.
Elements of the threshold vector $\mathbf{T}_1$ are determined based on the posterior error probabilities computed by using the parity-checks specified by Definition 1. We assume that for each codeword bit, the parity-checks used are orthogonal on that bit, meaning that except for that bit, every other involved unknown bit appears in exactly one of the parity-checks. Finally, assuming as an appropriate approximation, that all the parity-check equations involve exactly two unknown

bits beside the considered one, for any $i = B+1, B+2, ..., L$, the threshold $T_1(i)$ is equal to the smallest integer such that the following inequality holds:

$$\frac{p}{1-p}\left(\frac{1+(1-2p)^2}{1-(1-2p)^2}\right)^{|\Omega_i|-2T_1(i)} \leq 1 \ , \tag{7}$$

where $|\Omega_i|$ denotes the number of parity-check equations related to the i-th information bit [10].

4.2 Iterative Decoding Algorithm - IDA

For a given $N^* \leq N$, IDA decodes the received sequence $[z_1, z_2, ..., z_{N^*}]$ for the (N^*, L) punctured simplex code employing a BP based bit-flipping (BP-BF) iterative decoding and the sets Ω_n^* of parity-check equations, specified by Definition 2, $n = L+1, L+2, ..., N^*$.

BP-BF based iterative decoding (see [4], for example) includes the following main difference in comparison with simple BF.

– For each bit n, and each combination of $|\Omega_n^*| - 1$ parity-checks out of the $|\Omega_n^*|$ parity checks associated with bit-n, make $|\Omega_n^*|$ estimate of the nth bit value associated with these combinations.

Accordingly, we employ the following iterative BP-BF based decoding algorithm.

– *Initialization*: $\hat{x}_n = z_n$ and $\hat{x}_{nm} = z_n$.
– *Iterative Processing*
  1. *Step 1*:
     (a) For each n and for each $m \in \Omega_n^*$, evaluate:
     $\sigma_n(m) = \sum_{n' \in \omega(m)} \hat{x}_{n'm}$ $[mod 2]$.
     (b) If all $\sigma_n(m) = 0$ go to Step 3 (a). If some maximum number of iterations (e.g. 30) is exceeded go to Step 3 (b).
  2. *Step 2*: For each n, do the following:
     (a) If $\sum_m^{|\Omega_n^*|} \sigma_n(m) \geq |\Omega_n^*|/2$, then $\hat{x}_n = \hat{x}_n \oplus 1$.
     (b) If $\sum_{m'}^{|\Omega_n^* \setminus m|} \sigma_n(m') \geq |\Omega_n^* \setminus m|/2$, then $\hat{x}_{nm} = \hat{x}_{nm} \oplus 1$.
     If no complementation was performed go to Step 3 (b); otherwise go to Step 1.
  3. *Step 3*:
     (a) $\hat{\mathbf{x}} = [\hat{x}_n]$ is the decoding result.
     (b) Algorithm halts and a warning is declared that a valid decoding is not reached.

5 Performance and Complexity

5.1 Performance

The performance of the novel algorithm is experimentally considered when the LFSR characteristic polynomial is chosen as $1+u+u^3+u^5+u^9+u^{11}+u^{12}+u^{17}+$

Table 1. Performance of the novel algorithm - experimental analysis: Error-rate of the LFSR initial state reconstruction, as a function of the correlation noise p when the LFSR length is $L = 40$, the characteristic polynomial weight is 17, and the length of the sequence available for processing is $N = 40000$ bits.

p	Error rate of LFSR initial state reconstruction					
	OSDA $B = 18$	OSDA $B = 19$	OSDA $B = 20$	OSDA $B = 21$	OSDA $B = 22$	IDA $N^* = 4096$, $B = 22$
0.25	0.009	0.000	0.000	0.000	0.000	0.000
0.26	0.015	0.000	0.000	0.000	0.000	0.000
0.27	0.024	0.000	0.000	0.000	0.000	0.000
0.28	0.081	0.000	0.000	0.000	0.000	0.000
0.29	0.159	0.006	0.000	0.000	0.000	0.000
0.30	0.254	0.023	0.000	0.000	0.000	0.000
0.31	0.384	0.041	0.002	0.000	0.000	0.000
0.32	0.569	0.098	0.002	0.000	0.000	0.000
0.33	0.696	0.226	0.020	0.000	0.000	0.000
0.34	0.838	0.356	0.053	0.001	0.000	0.000
0.35	0.915	0.542	0.114	0.002	0.001	0.000
0.36	0.955	0.743	0.225	0.019	0.022	0.000
0.37	0.983	0.865	0.450	0.080	0.062	0.001
0.38	0.990	0.932	0.652	0.210	0.208	0.023
0.39	0.997	0.980	0.850	0.445	0.399	0.052
0.40	1.000	0.988	0.935	0.663	0.651	0.267

$u^{19}+u^{21}+u^{25}+u^{27}+u^{29}+u^{32}+u^{33}+u^{38}+u^{40}$ and $N = 40000$ (i.e. assuming the same example as was considered in [8]-[9]). Note that the proposed algorithm can be applied for values of L significantly longer than $L = 40$, but this value was employed in all numerical and experimental illustrations for comparison with previously reported results.

Results of the performance analysis are presented in Table 1. This table displays the error-rate of the LFSR initial state reconstruction as a function of the correlation noise p when the algorithm employs:
(i) OSDA with B = 18, 19, 20, 21, 22,
(ii) IDA with $N^* = 4096$, $B = 22$, and at most 20 iterations.

Each error-rate given in the table is obtained by calculation over a corresponding, randomly selected, set of 1000 samples. Recall that "error-rate" indicates the fraction of trials for which we obtain incorrect decoding (and accordingly incorrect reconstruction of the secret key).

5.2 Complexity

Recall that the overall complexity assumes time and space complexity requirements. The complexity analysis yields that according to the structure of the proposed algorithm:
- The algorithm requires a space for the input. Space requirements for the decoding process are as follows: when OSDA is employed, decoding processing does not require memory; IDA requires a memory proportional to the parameter N^*;
- Time complexity is specified by the following corollaries.

Corollary 1. Assuming that OSDA is employed, that $|\Omega|$ denotes the average cardinality of the parity-check sets $|\Omega_i|$, that ω denotes the average number of bits in a parity-check, and that w denotes the weight of the LFSR characteristic polynomial, the implementation complexity of the proposed algorithm is proportional to $2^B[\ (L-B)|\Omega|\omega\ +\ (N-L)w\ ]\ mod2$ additions.

Corollary 2. Assuming that IDA is employed, that $|\Omega^*|$ denotes the average cardinality of the parity-check sets $|\Omega_n|$, that ω^* denotes the average number of bits in a parity-check, that I denotes the number of iterations, and that w denotes the weight of the LFSR characteristic polynomial, the implementation complexity of the proposed algorithm is proportional to $2^B[\ I(N^*-L)|\Omega^*|(|\Omega^*|-1)\omega^*\ +\ (N-L)w\ ]\ mod2$ additions.

Note also that from the structure of the proposed algorithms, it is readily seen that the proposed algorithms are suitable for fast software implementation, as well as for simple hardware implementation: the algorithms employ only simple arithmetic operations ($mod2$ addition) and simple logical operations.

Also, since the decoding process is mainly memoryless, note that a reduction of the time complexity specified by the previous corollaries can be obtained by an appropriate time-memory complexity trade-off.

Finally note that in the presented experiments, the decoding step has employed the underlying codeword lengths $N = 40000$ and $N^* = 4096$ for OSDA and IDA, respectively. This is an illustration that OSDA and IDA yield a trade-off between the length of the required sample (i.e. the code rate) and the decoding complexity.

6 Comparison of the Novel Algorithm with Recently Proposed Improved Fast Correlation Attacks

This section presents an arguably comparative analysis of the underlying principles, performance and complexity of recently proposed improved fast correlation attacks [9] and the novel algorithm, assuming the same input.

6.1 Comparison of the Underlying Principles

Comparison of the underlying principles employed in [8]-[9] and in the novel algorithm for the fast correlation attack can be summarized as follows.

– The approaches of [8]-[9] are based on decoding of convolutional codes and turbo codes with convolutional codes as the component codes constructed over the LFSR sequence. The novel approach is based on decoding punctured simplex block codes corresponding to the LFSR sequence.
– The algorithms [8]-[9] and the novel algorithm employ different parity-checks. The parity-checks employed in [9]-[8] are constructed by searching for these parity checks which include the following bits: currently considered bit, bits from a subset of B previous bits, and no more than two other bits.
  The parity-checks employed in the novel algorithm are constructed by searching for these parity checks which include the following bits:
  (i) currently considered information bit, bits from a subset of B first information bits, and two other information bits with the corresponding parity-bit, or two arbitrary parity bits only, or
  (ii) currently considered parity bit, bits from a subset of B first information bits, and no more than two other parity bits.
  Note that these different approaches in the parity-check constructions imply different number of parity-checks per bit, as well.
– The decoding techniques employed in [8]-[9] are Viterbi decoding, BCJR decodings, and MAP turbo decoding (see [7] and [1]). On the other hand the novel algorithm employs the following two low-complexity decoding techniques: (i) APP threshold decoding, and (ii) BP based BF iterative decoding.
– The fast correlation attacks from [8]-[9] implicitly include an exhaustive search over a set of dimension 2^B through employment of the Viterbi or BCJR decodings due to the trellis search. The novel algorithm employs an explicit search over all 2^B possible patterns corresponding to the first B information bits.
– A decoding process based on the Viterbi or BCJR algorithm requires a memory of dimension proportional to 2^B. On the other hand, OSDA does not require memory, and IDA requires a memory proportional to the parameter N^*.

6.2 Comparison of the Performance and Complexity

For the performance comparison of the novel and turbo based fast correlation attacks [9] the same inputs are employed and relevant parameters are selected so that the novel algorithm always has significantly lower overall implementation complexity than the algorithm [9].

According to [9], the time complexity of the turbo decoding is proportional to $2^B IMJm$ real multiplications where I denotes the number of the iterations,

Table 2. Comparison of the algorithms performance, assuming the same inputs, and lower complexity of the novel algorithm in comparison to the turbo algorithm [9]: Limit noise for which the algorithms yield, with probability close to 1, correct reconstruction of the initial LFSR state, when the LFSR characteristic polynomial is $1 + u + u^3 + u^5 + u^9 + u^{11} + u^{12} + u^{17} + u^{19} + u^{21} + u^{25} + u^{27} + u^{29} + u^{32} + u^{33} + u^{38} + u^{40}$, and the available sample is 40000 bits.

ALGORITHM	Limit Noise
turbo algorithm [9]: $B = 15$, $M = 2$	0.27
novel algorithm with OSDA: $B = 19$	0.28
turbo algorithm [9]: $B = 15$, $M = 4$	0.29
novel algorithm with OSDA: $B = 21$	0.33
turbo algorithm [9]: $B = 15$, $M = 16$	0.30
novel algorithm with OSDA: $B = 22$	0.34
novel algorithm with IDA: $N^* = 4096$, $B = 22$	0.36

M the number of the component codes, J the number of processed bits, and m the number of employed parity-checks per bit. The time complexity of the novel algorithm is given in Corollaries 1 and 2.

Also note that the space complexity of the approach from [9] is proportional to 2^B due to employment of the BCJR algorithm. If OSDA is employed no space complexity is required, and if IDA is employed it is usually significantly smaller than 2^B due to its linear rather than exponential nature.

An illustrative performance comparison is presented in the Table 2.
Note that, in each case, the complexity of the proposed algorithm could be considered as significantly lower than complexity of the turbo decoding [9] although the proposed algorithm assumes search over a much larger set of hypotheses, since: (i) [9] employs iterative processing with M component codes and (ii) the dominant arithmetic operation in the proposed algorithm is $mod2$ addition against real multiplication for the turbo based decoding of [9].

Finally, note that the actual time for performing the attack by the novel algorithm strongly depends on the implementation constraints so that a straightforward comparison is not appropriate. Also, the approaches of [8]-[9] can be modified to involve $mod2$ additions, but at the expense of performance degradation.

7 Conclusions

A novel algorithm for the fast correlation attack has been proposed. The algorithm is based on decoding procedures of the corresponding binary block code with novel constructions of the parity-checks, independent of the LFSR feedback polynomial weight, and the following two decoding approaches are employed: an APP based threshold decoding and a BP based BF iterative decoding . The constructions of the parity-checks are based on searching for certain parity-check equations and their linear combinations employing the finite-state machine model of an LFSR with primitive characteristic polynomial. The expected numbers of the parity-checks per parity bit have been derived, showing that a large number of appropriate parity-checks can be constructed.

The performance of the proposed algorithm has been analyzed experimentally showing that the algorithm is a powerful one.

The overall implementation complexity has been specified. As dominant operations the algorithm employs *mod*2 additions and simple logical operations, so that it is very suitable for high-speed software implementation as well as for simple hardware implementation.

The algorithm offers good trade-offs between required sample length (i.e. rate of the underlying code), overall complexity and performance. The one-step threshold decoding approach yields high performance assuming long enough sample, and the iterative decoding approach can reach the same performance using a significantly shorter sample but at the expense of increased complexity.

The novel algorithm has been compared with recently reported improved fast correlation attacks based on convolutional codes and turbo decoding. The underlying principles, performance and complexity have been compared, and the essential gain obtained with the novel approach is pointed out. The developed algorithm has the following two main advantages over other reported ones:
(a) Assuming a lower overall complexity, and the same inputs, the proposed algorithm yields significantly better performance.
(b) It is suitable for high-speed software implementation as well as for simple hardware implementation and highly parallel architectures.

References

1. L. Bahl, J. Cocke, F. Jelinek and J. Raviv, “Optimal decoding of linear codes for minimizing symbol error rate,” *IEEE Transactions on Information Theory*, vol. IT-20, pp. 284-287, March 1974.
2. V. Chepyzhov and B. Smeets, "On fast correlation attack on certain stream ciphers", Advances in Cryptology - EUROCRYPT '91, *Lecture Notes in Computer Science*, vol. 547, pp. 176-185, 1991.
3. A. Clark, J. Dj. Golić, and E. Dawson, "A comparison of fast correlation attacks," Fast Software Encryption - FSE'96, *Lecture Notes in Computer Science*, vol. 1039, pp. 145-157, 1996.
4. R. G. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. IT-8, pp. 21-28, Jan. 1962.
5. M. P. C. Fossorier, M. J. Mihaljević and H. Imai, "Reduced complexity iterative decoding of Low Density Parity Check codes based on Belief Propagation", *IEEE Transactions on Communications*, vol. 47, pp. 673-680, 1999.

6. M. P. C. Fossorier, M. J. Mihaljević and H. Imai, "Critical noise for convergence of iterative probabilistic decoding with belief propagation in cryptographic applications", Applied Algebra, Algebraic Algorithms and Error Correcting Codes - AAECC 13, *Lecture Notes in Computer Science*, vol. 1719, pp. 282-293, 1999.
7. R. Johannesson and K. Zigangirov, *Fundamentals of Convolutional Coding.* New York: IEEE Press, 1999.
8. T. Johansson and F. Jonsson, "Improved fast correlation attacks on stream ciphers via convolutional codes", Advances in Cryptology - EUROCRYPT'99, *Lecture Notes in Computer Science*, vol. 1592, pp. 347-362, 1999.
9. T. Johansson and F. Jonsson, "Fast correlation attacks based on turbo code techniques", Advances in Cryptology - CRYPTO'99, *Lecture Notes in Computer Science*, vol. 1666, pp. 181-197, 1999.
10. J. L. Massey, *Threshold Decoding.* Cambridge, MA: MIT Press, 1963.
11. A. Menezes, P.C. van Oorschot and S.A. Vanstone, *Handbook of Applied Cryptography.* Boca Raton: CRC Press, 1997.
12. W. Meier and O. Staffelbach, "Fast correlation attacks on certain stream ciphers," *Journal of Cryptology*, vol. 1, pp. 159-176, 1989.
13. M. J. Mihaljević and J. Dj. Golić, "A fast iterative algorithm for a shift register initial state reconstruction given the noisy output sequence", Advances in Cryptology - AUSCRYPT '90, *Lecture Notes in Computer Science*, vol. 453, pp. 165-175, 1990.
14. M. J. Mihaljević and J. Dj. Golić, "A comparison of cryptanalytic principles based on iterative error-correction," Advances in Cryptology - EUROCRYPT '91, *Lecture Notes in Computer Science*, vol. 547, pp. 527-531, 1991.
15. M. J. Mihaljević, M.P.C. Fossorier and H. Imai, "Novel fast correlation attack via iterative decoding of punctured simplex code", Proceedings of *IEEE ISIT'2000*, Sorento, Italy, June 2000.
16. M. J. Mihaljević and J. Golić, "A method for convergence analysis of iterative probabilistic decoding", accepted for publication in *IEEE Transactions on Information Theory.*
17. W. Penzhorn, "Correlation attacks on stream ciphers: Computing low-weight parity checks based on error-correcting codes", Fast Software Encryption - FSE'96, *Lecture Notes in Computer Science*, vol. 1039, pp. 159-172, 1996.
18. T. Siegenthaler, "Decrypting a class of stream ciphers using ciphertext only", *IEEE Transactions on Computers*, vol. C-34, pp. 81-85, 1985.
19. K. Zeng and M. Huang, "On the linear syndrome method in cryptanalysis," Advances in Cryptology - CRYPTO '88, *Lecture Notes in Computer Science*, vol. 403, pp. 469-478, 1990.
20. K. Zeng, C.H. Yang and T.R.N. Rao, "An improved linear syndrome method in cryptanalysis with applications," Advances in Cryptology - CRYPTO '90, *Lecture Notes in Computer Science*, vol. 537, pp. 34-47, 1991.