On A Fast Correlation Attack on Certain Stream Ciphers

Vladimir Chepyzhov
Institute for Problems of Information Transmission
USSR Academy of Sciences
Ermolovoy 19, Moscow GSP-4, USSR

Ben Smeets
Department of Information Theory
Lund University
Box 118
S-221 00 Lund, Sweden

Abstract—In this paper we present a new algorithm for the recovery of the initial state of a linear feedback shift register when a noisy output sequence is given. Our work is focussed on the investigation of the asymptotical behaviour of the recovery process rather than on the construction of an optimal recovery procedure. Our results show the importance of low-weight checks and show also that the complexity of the recovery problem grows less than exponentially with the length of the shift register, even if the number of taps grows linearly with the register length. Our procedure works for shift register with arbitrary feedback polynomial.

1 Introduction

It was observed by Siegenthaler [1] that if the key generator used in a stream cipher is correlated to a linear feedback shift (LFSR) sequence with a probability that exceeds 0.5, then it is possible possible to reconstruct the initial state of the shift register. In the traditional setting it is assumed that the feedback connections of the LFSR are known to the cryptanalyst and that only the initial state is unkown to him. In the correlation attack as it originally was desribed by Siegenthaler, one uses an exhaustive search through the state space of the shift register. Such a search is not very realistic when the degree r (= length of the LFSR) of the feedback polynomial of the LFSR exceeds 60, especially when this task has to be performed frequently. Recently it was shown by Meier and Staffelbach [2] that in certain cases one can avoid this exhaustive search. In particular they showed that if the number t of feedback taps is small, then it is possible to restore the initial state by an iterative procedure with much less complexity than exhaustive search. Using the same idea, several others authors proposed iterative improvement algorithms, see for example [3]. Unfortunately the

D.W. Davies (Ed.): Advances in Cryptology - EUROCRYPT '91, LNCS 547, pp. 176-185, 1991.

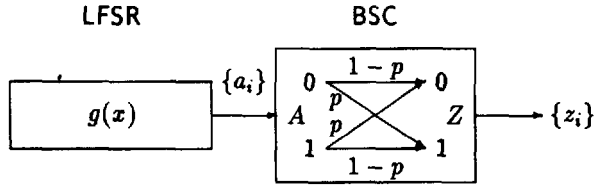

Figure 1: The correlation attack model for initial state recovery problem.

length of the key sequence that must be available to the cryptanalyst may be forbidding large. The algorithm of Meier and Staffelbach has asymptotical complexity $O(r)$ when the number of taps t is fixed. However, if t grows linearly with r, then the complexity of their algorithm is exponential in r.

In this paper we present another procedure for finding the initial state of the LFSR. The algorithm is divided into two stages; the first stage is the search for low-weight checks and the second is the iterative improvement procedure for restoring the initial state. Also in the algorithm of Meier and Staffelbach (and others too) there are two stages but there the required checks are obtained by a simple algebraic procedure. Their procedure, albeit very fast, uses the available key stream not very efficiently and thus forces the cryptanalyst to exhibit long key stream subsequences. Our algorithm is using the key stream almost as efficiently as possible at the expense of an increase of the complexity of the first stage. Our algorithm that we use for the first stage is derived from efficient algorithms for finding the non-zero codeword of lowest weight in a linear code, [4], [5]. The second stage of our algorithm is almost identical to Gallager's algorithm for the decoding of low-density parity-check codes [6].

2 The cryptanalyst's problem and definitions

Our attack is like in [1] and [2] based on the model shown in Figure 1. The (observed) output sequence $\{z_i\}$ is regarded as a corrupted version of the LFSR output sequence $\{a_i\}$. It is assumed that the value of z_i only depends on the input a_i and the observation at time i. For this model it is natural to say that the symbol a_i is passed through an (observation) channel whose output z_i we can observe. The model of this channel we just described is also known as the Binary (Memoryless) Symmetric Channel, BSC. The probability $q = 1 - p$ is also known as the correlation probability since $\mathrm{P}(a_i = z_i) = q$. The cryptanalyst's problem can now be formulated as follows:

Statement of the problem: Let $p < 0.5$ an let the (primitive)[1] feedback polynomial of the LFSR, denoted by $g(x)$, be of degree r. The problem is to restore the linear feedback shift register subsequence $\{a_i\}_{i=0}^{N-1}$ from the observed subsequence $\{z_i\}_{i=0}^{N-1}$, where $r < N << 2^r - 1$. The value of N should be as small as possible.

[1] For simplicity the reader can think of primitive feedback polynomial. However, mutatis mutandis, everything is valid for arbitrary feedback polynomials.

We fix N to some value greater than r. Later we shall see how small a value of N can be chosen. This value of N is lower bounded by the unicity distance given by $r/(1-H(p))$, where $H(x)$ is the binary entropy function $H(x) \doteq -x\log_2 x - (1-x)\log_2(1-x)$. Here we assumed that all possible initial states are equally likely. Values of N close to this lower bound can be obtained att the expense of an exhaustive search through the initial state space as is illustrated by the results in [1]. But this gives an exponential growth (with r) in complexity of the recovery procedure. We want to avoid such a growth in complexity.

Before we start with the discussion of our procedure, we give some definitions. As in [2] the following polynomials play an essential role:

Definition 1: Let $h(x)$ be a polynomial of degree $\geq r = \deg g(x)$, with $h_0 = h(0) = 1$ such that for any sequence $\{a_i\}$ (whose formal power series we denote by $a(x)$) generated by the LFSR with feedback polynomial $g(x)$ we have $h(x)\cdot a(x)$ is a polynomial of degree $< \deg h(x)$. Then the polynomial $h(x)$ is called *a check* (polynomial of the LFSR).

From shift register theory we know that whenever the polynomial $h(x)$ is a check, then $g(x)$ divides $h(x)$. From another point of view we can regard the checks of degree less than n as the codewords of the $(n, n-r)$ linear code generated by the polynomial $g(x)$. This code has rate $R = \frac{n-r}{n}$ and redundancy r. To simplify the exposition of our attack we assume the we already obtained a set of M checks $\{h^{(i)}(x)\}_{i=1}^{M}$.

Definition 2: Let the set $\{h^{(i)}(x)\}_{i=1}^{M}$ be a set of checks and let the coefficients $h_j^{(i)}$ be determined from $h^{(i)}(x) = \sum_{j=0}^{\infty} h_j^{(i)} x^j$. This set is called *a set of M checks of length L with at most t taps* if:

1. $\forall i = 1, \ldots, M, \quad h_0^{(i)} = 1$, and $\deg h^{(i)} < L$,
2. $\forall h^{(i)}(x), \quad \#\left\{h_j^{(i)};\, h_j^{(i)} \neq 0,\, j \geq 1\right\} \leq t$,
3. $\forall \ell_1, \ell_2 : \ell_1 \neq \ell_2, \quad \left\{j;\, h_j^{(\ell_1)} \neq 1\right\} \cap \left\{j;\, h_j^{(\ell_2)} \neq 1\right\} = \emptyset$.

In words this means that the Hamming weight of the polynomials in the set of checks is not more than $t+1$ and that two different checks have different non-zero coefficients except for their constant term.

Example: The following polynomials are independent checks of length L with at most t taps in the case $g(x) = 1 + x$, $t = 3$ and $L = 8$.

$$h^{(1)}(x) = 1 + x + x^5 + x^7,$$
$$h^{(2)}(x) = 1 + x^2 + x^3 + x^4,$$
$$h^{(3)}(x) = 1 + x^6.$$

3 Iterative improvement

Let us consider some symbol a_i from the sequence $\{a_i\}$. If $h(x) = \sum_{j=0}^{L-1} h_j x^j$ is a check, then

$$a_i = \sum_{j \geq 1, h_j = 1} a_{i-j} = a_{i_1} + a_{i_2} + \ldots + a_{i_t},$$

where the i_k's denote the value of the indices $i-j$ when $h_j = 1$. If we have a set of M independent checks $\{h^{(i)}\}_{i=1}^{M}$, then we have for this symbol a_i

$$\begin{aligned} a_i + a_{i_1^{(1)}} + a_{i_2^{(1)}} + \cdots + a_{i_t^{(1)}} &= 0 \\ a_i + a_{i_1^{(2)}} + a_{i_2^{(2)}} + \cdots + a_{i_t^{(2)}} &= 0 \\ \vdots \qquad\qquad \vdots \qquad\qquad & \vdots \\ a_i + a_{i_1^{(M)}} + a_{i_2^{(M)}} + \cdots + a_{i_t^{(M)}} &= 0. \end{aligned} \tag{1}$$

We note that all the indices $i_j^{(\ell)}$ are different. It is easy to see that these equations can be applied to every symbol a_i as long as $i - L > 0$. If we replace in equation (1) the values of a_i and $a_{i_j^{(\ell)}}$ by the corresponding values of z_i and $z_{i_j^{(\ell)}}$, then maybe not all the equations hold because the a_i's are not necessarily equal to the corresponding z_i's.

Using these (check) equations it is possible to compute a new value $\tilde{z}_i^{(1)}$ of z_i such that $\mathrm{P}(\tilde{z}_i^{(1)} = a_i) > q$ using the ideas of Gallager for the decoding of low-density parity-check codes. We compute for every index i a new value of z_i and we obtain the (sub)sequence $\{\widetilde{z^{(1)}}_i\}_{i=0}^{N-1}$. Using the procedure again on this sequence we can get the sequence $\{\widetilde{z^{(2)}}_i\}_{i=0}^{N-1}$ in such a manner that the number of disagreements between $\{\widetilde{z^{(s)}}_i\}_{i=0}^{N-1}$ and $\{z_i\}_{i=0}^{N-1}$ vanishes as s increases. Our procedure obviously works under the conditions that the first step must lead to a smaller number of expected disagreements. Hence the first step deserves our special attention.

In what follows we consider the noisy output symbols as randomvariables, conveniently denoted by Z_i, $i \geq 0$. We define the random variable

$$\tilde{Z}_i \stackrel{\text{def}}{=} \sum_{j=1}^{t} h_j Z_{i-j}. \tag{2}$$

When no errors occur, then $Z_i = \tilde{Z}_i = a_i$. Therefore we also consider the random variable

$$Y_i \stackrel{\text{def}}{=} Z_i + \tilde{Z}_i, \tag{3}$$

and we note that $Y_i = 0$ when $Z_i = a_i$ for all i. The probability $\mathrm{P}(Z_i = a_i) = q = 1 - p > 0.5$. The probability that $\tilde{Z}_i = a_i$ is given by

$$\begin{aligned} \mathrm{P}(\tilde{Z}_i = a_i) &= \mathrm{P}(\#\text{symbol changes is even}) \\ &= \sum_{i=0, i\,even}^{t} \binom{t}{i} p^i q^{t-i} = 1 - \sum_{i=0, i\,odd}^{t} \binom{t}{i} p^i q^{t-i} \\ &= \frac{1}{2}\left[\sum_{i=0}^{t} \binom{t}{i} p^i q^{t-i} - \sum_{i=0}^{t} \binom{t}{i} (-p)^i q^{t-i}\right] \\ &= \frac{1}{2}[1 + (2\varepsilon)^t]. \end{aligned} \tag{4}$$

Let

$$Q \stackrel{\text{def}}{=} \mathrm{P}(\tilde{Z}_i = a_i). \tag{5}$$

Let D (random variable !) denote the number of non zeros in $\{Y_i\}_{i=0}^{N-1}$. In the first correction step we determine the sequence $\left\{s^{(1)}{}_i\right\}$ defined by

$$z_i^{(1)} = \begin{cases} z_i, & \text{when } D < T \\ z_i + 1, & \text{when } D \geq T. \end{cases}$$

We now compute the probabilities $\mathrm{P}(z_i^{(1)} = a_i)$ after the first correction of $\{z_i\}$. then

$$\mathrm{P}(z_i^{(1)} = a_i) = \mathrm{P}(z_i^{(1)} = a_i, D < T) + \mathrm{P}(z_i^{(1)} = a_i, D \geq T),$$

where T is some fixed integer value in the range $[0, M]$ to be chosen later. Since

$$\begin{aligned} \mathrm{P}(z_i^{(1)} = a_i, D < T) &= \mathrm{P}(z_i^{(0)} = z_i = a_i, D < T) \\ &= q \sum_{d=0}^{T-1} \binom{M}{d} (1-Q)^d Q^{M-d}, \end{aligned} \tag{6}$$

and, similarly,

$$\mathrm{P}(z_i^{(1)} = a_i, D \geq T) = p \sum_{d=T}^{M} \binom{M}{d} Q^d (1-Q)^{M-d}. \tag{7}$$

Combining these two results gives

$$\mathrm{P}(z_i^{(1)} = a_i) = q + \sum_{d=T}^{M} \binom{M}{d} (Q(1-Q))^{M-d} \left[pQ^{2d-M} - q(1-Q)^{2d-M} \right]. \tag{8}$$

We get on the average more correct symbols after the first step if

$$\tilde{Q} \stackrel{\text{def}}{=} \mathrm{P}(z_i^{(1)} = a_i) \geq q = \mathrm{P}(z_i = a_i). \tag{9}$$

By inspection of equation (8) it follows that if

$$pQ^{2T-M} - q(1-Q)^{2T-M} > 0,$$

then for all $i \geq T$

$$pQ^{2i-M} - q(1-Q)^{2i-M} > 0.$$

Thus we have the following Lemma:

Lemma 1: Let Q and $\tilde{Q}$ be defined by equation (5), resp. equation (9). Furthermore, let

$$T = \max \left\{ d \in [0..M]; \left(\frac{Q}{1-Q} \right)^{M-2d} > \frac{q}{1-q} \right\}, \tag{10a}$$

then

$$\tilde{Q} > Q \quad \text{if and only if} \quad \left(\frac{Q}{1-Q} \right)^{M} > \frac{q}{1-q}. \tag{10b}$$

∎

From this we have the following important result:

Theorem 2: For convergence of the iterative improvement procedure to the correct (sub)sequence $\{a_i\}$ we need at least

$$M = \frac{\log \frac{q}{1-q}}{\log \frac{Q}{1-Q}} \tag{11}$$

independent checks. ∎

We have the corollary:

Corollary 3: Let $q = 1/2 + \varepsilon$ and $Q = \frac{1}{2}(1 + (2\varepsilon)^t)$, then we need

$$M_0 = O\left(\left(\frac{1}{2\varepsilon}\right)^{t-1}\right) \quad \text{checks.}$$

∎

When the conditions for convergence are satisfied we know from [6] that the required number of iterations is normally very small. Hence, by ensuring that we meet the conditions of convergence with some additional safety margin, the iterative improvement process has roughly the complexity $\mathcal{C}_{it} = O(MN)$ since for all the N symbols we have to evaluate the M check equations. Summarizing we see that the fewer the taps t, the fewer checks we need.

4 Finding low-weight checks

The value of N that can be chosen is related with the problem of how many symbols of $\{z_i\}$ we need for obtaining the set of M independent checks. This brings us back to the first stage of our attack procedure: the search of non-zero codewords of low (Hamming) weight. If these codewords have large degrees, then we need also a large N because we want only those codewords which give independent checks.

Recall that the linear code we have to investigate has length n, informativity $n-r$, and redundancy r. We want a fast algorithm that will find the non-zero codewords of small weight, possibly the lowest. This algorithm should be applicable to any linear code and should have small complexity for almost all linear codes. The following lemma is important:

Lemma 4: For virtually all linear (n, k) codes having a minimum distance at most d, the minimum weight codeword ($\neq 0$) can be found in

$$O\left(2^{H(\frac{d}{n})k}\right) \text{ steps}$$

using the algorithm described below. ∎

By virtually we mean that the result holds for all but a fraction of codes which vanishes when n goes to infinity.

The lemma follows from the results in [4] and [5]. We give a sketch of the proof using some probabilistic argument.

Sketch of the proof: Let us consider a linear binary (n, k) code with generating matrix G. Note that this is an $n \times k$ binary matrix. Let us also consider a fixed information set of k digits of this code. (An information set is a set of indices of k columns of G that have full rank k.) If the minimum codeword has weight d, then with high probability the informational part of this word has weight at most $d \cdot \frac{k}{n}$. To find this word we have to try all possible patterns of weight at most $d \cdot \frac{k}{n}$ as the informational part. Each of these patterns has a corresponding redundancy part (of the codeword). For each pattern we check whether the sum of the weights of the informational and redundancy parts is at most d. We continue with this process until we find a minimal codeword. The complexity of this algorithm is of the same order as the number of patterns of length k and of weight at most $d\frac{k}{n}$. i.e.,

$$\sum_{0 \le i < d\frac{k}{n}} \binom{k}{i} \le 2^{H(\frac{d}{n})k}.$$

Using Gray codes this trial process can be implemented efficiently. □

Remark: By using at most n different informational sets we can avoid the probabilistic nature of the proof. The extra cost of this has no influence on our final result.

From coding theory we know that virtually all linear (n, k) codes of fixed rate $R = k/n$ satisfy the Varshamov-Gilbert bound: $d = n \cdot H^{-1}(1-R) + o(n)$. Thus we can expect that the complexity of finding a codeword of weight $d = n \cdot H^{-1}(1-R) + o(n)$ has order $O(2^{(1-R)k}) = O(2^{(1-\frac{r}{n})r})$. Combining all this we arrive at the following result.

Lemma 5: The complexity of finding a set of M independent checks with at most $d-1$ taps and length $(d-1)M + n$ is of order

$$C_{chk} = M \times O(2^{(1-\frac{r}{n})r}),$$

where $d = n \cdot H^{-1}(1-R) + o(n)$. ■

Proof: The cost of finding the first check of length at most n is $O(2^{(1-\frac{r}{n})r})$. For the second check we have to cancel the $d-1$ coefficient positions occupied by the first check. In order to search again among n possible positions we add $d-1$ to the allowed length of the second check. Hence, after M checks we have a length that is not more than $M(d-1) + n$. The search complexity for every check is essentially the same. Thus the total cost is $M \times O(2^{(1-\frac{r}{n})r})$. □

We are now ready to formulate the main result:

Theorem 6: Let $p = \mathrm{P}(Z_i \neq a_i)$, $p = \frac{1}{2} - \varepsilon < \frac{1}{2}$, and let us fix $n > r$ and $d = n \cdot H^{-1}(1-R) + o(n)$. To break the system we need to observe a segment of length N of $\{z_i\}$, where

$$N = 2((d-1)M + n). \tag{12a}$$

Furthermore, we need

$$M = O\left(\left(\frac{1}{2\varepsilon}\right)^d\right), \quad \left(M \approx 2M_0, \text{ where } M_0 = \left(\frac{1}{2\varepsilon}\right)^{d-2}\right). \tag{12b}$$

The first stage of the algorithm has complexity

$$\mathcal{C}_{chk} = O\left(\left(\frac{1}{2\varepsilon}\right)^d \cdot 2^{(1-\frac{r}{n})r}\right) \tag{13a}$$

and the second stage for recovering the initial state has complexity

$$\mathcal{C}_{it} = O\left(d\left(\frac{1}{2\varepsilon}\right)^{2d}\right). \tag{13b}$$

■

Proof: In order to break the system we determine first twice as many checks than is given by Corallary 3. The maximal length of this set is given by Lemma 5 and is not more than $(d-1)M+n$. With this set we can perform the first step on the first halve of the known $2((d-1)M+n)$ z_i symbols. The second halve of the symbols are dealt with by using the reciprocals (or reversed forms) of the polynomials in the set. Thus we can carry out the first step if we have $2((d-1)M+n)$ symbols of $\{z_i\}$. Convergence in the first and subsequent iteration is guaranteed by Theorem 2. The results on the complexities are immediate from the previous results. □

The total work load is of course the sum $\mathcal{C}_{alg} = \mathcal{C}_{chk} + \mathcal{C}_{it}$ and we have to choose the parameters n and d in such a way that we minimize $\mathcal{C}_{alg}$. For this purpose fix $\rho = \frac{r}{n}$ and let r go to infinity. Then $\delta \stackrel{\text{def}}{=} \lim_{r\to\infty} \frac{d}{r} = \frac{H^{-1}(\rho)}{\rho}$, i.e., $H(\delta\rho) = \rho$. We introduce

$$\alpha(\rho) \stackrel{\text{def}}{=} \lim_{r\to\infty} \frac{\log_2 \mathcal{C}_{chk}}{r} = 1 - \rho + L\delta, \tag{14a}$$

$$\beta(\rho) \stackrel{\text{def}}{=} \lim_{r\to\infty} \frac{\log_2 \mathcal{C}_{it}}{r} = 2L\delta, \tag{14b}$$

where $L = -\log_2(2\varepsilon)$. Furthermore,

$$\gamma(\rho) \stackrel{\text{def}}{=} \lim_{r\to\infty} \frac{\log_2(\mathcal{C}_{chk} + \mathcal{C}_{it})}{r} = \max(\alpha(\rho), \beta(\rho)). \tag{14c}$$

For the optimal algorithm we have

$$\gamma_{opt}(\varepsilon) = \min_{0<\rho<1} \gamma(\rho).$$

Thus we obtain the following corollary to the previous theorem:

Corollary 7: The complexity to break the system has order $O\left(2^{\gamma_{opt}(\varepsilon)r}\right)$. ■

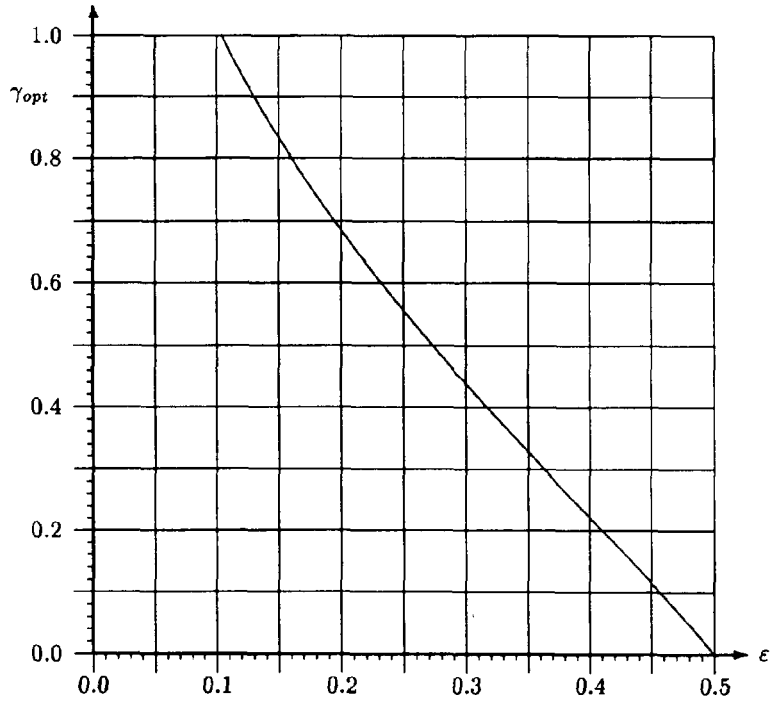

Figure 2: The function γ_{opt}.

In Figure 2 we show γ_{opt} as a function of ε. We see that our algorithm has a complexity less than $O(2^r)$ for $\varepsilon > 0.10$. In Reference [5] another algorithm is described for finding the minimal weight codeword(s). Using this algorithm we obtain that the overall complexity is less than $O(2^r)$ for all ε. However the result in [5] relies on the ability to solve a combinatorial problem for which a complexity bound can be derived but for which the corresponding algorithm is missing. For this reason we paid only attention to our simple algorithm. In any case, using the algorithm we have described, we find that $\gamma_{opt} = 0.56$ when $p = 0.25$ which we have for the generators of Geffe, Pless and Bruer.

5 Conclusion

Summarizing, we have obtained a new algorithm for breaking stream ciphers and we have given a detailed asymptotical analysis. It is important to mention that we deliberately avoided to use more clever iterative improvement schemes like the ones discussed in [2], [3], and [6], since this would make the asymptotic analysis harder, maybe impossible, and in any case less transparent. The analysis clearly shows the importance of checks of (a) low-weight and (b) of low degree. Property (a) leads to a smaller number of required checks and property (b) leads to a smaller demand of the number of known (or observed) symbols. The analysis also shows that the complexity

of the algorithm is less than exhaustive search even when t grows linearly with the length of the shift register.

We have verified our analysis through simulations. Since the second stage of the algorithm has been verified through the works of [2] and [6] we concentrated our simulations on the first stage. Our results show that we indeed obtain the required number of checks with an amount of work (=time) that is in accordence with Theorem 6. In addition we observed that we can get a set of M independent checks which is "dense". By this we mean that $|\{j\,;\, h_j^{(i)}, i = 1, 2, \ldots, M\}| \,/ \max_i \deg h^{(i)}(x) \approx 1$, or, in words, for almost all indices $j = 0, 1, \ldots, L-1$, L the length of the set of checks, there is a polynomial $h^{(i)}(x)$ with a coefficient that checks z_j.

6 Acknowledgement

The first author would like to thank the USSR Academy of Sciences, the Royal Swedish Academy of Sciences, and the Department of Information Theory in Lund for their support and making this work possible.

References

[1] T. Siegenthaler, "Decrypting a class of stream ciphers using ciphertext only", IEEE Trans. Comput., Vol. C-34, 1985, pp. 81-85.

[2] W. Meier, and O. Staffelbach, "Fast correlation attacks on certain stream ciphers", J. Cryptology, 1989, pp. 159-176.

[3] M. Mihaljevic, and J. Golic, "A fast iterative algorithm for a shift register initial state reconstruction given the noisy output sequence", *Proc. Auscrypt 1990*, pp. 165-175.

[4] G. S. Evseev, "Complexity of decoding for linear codes", Probl. Peredach. Inform., Vol. 19, 1983, pp. 3-8.

[5] J. T. Coffey, and R. M. Goodman, "The complexity of information set decoding", IEEE Trans. Inform. Theory, Vol. IT-36, 1990, pp. 1031-1037.

[6] R. G. Gallager, *Low-Density Parity-Check Codes*, MIT Press, Cambridge, MA, 1963.