Error Decomposition Algorithm for Approximating the k-Error Linear Complexity of Periodic Sequences

Fangwen Yu, Ming Su*, Gang Wang, Mingming Ren
Nankai-Baidu Joint Lab, Department of Computer Science, Nankai University, China

***Abstract*—In cryptography applications, pseudorandom sequences should have large linear complexity and k-error linear complexity, so that they cannot be recovered by only knowing a small amount of consecutive terms. However, general efficient algorithms do not exist for computing the exact value of k-error linear complexity. Therefore, it is useful to compute a good upper bound of the k-error linear complexity. First we develop an efficient exhaustive search algorithm for k-weight complexity when k is small by using Blahut's theorem and the cyclotomic structure of the discrete Fourier transform (DFT) of periodic sequences. Then we give an approximation algorithm by decomposing the total k errors into errors of smaller weight at different stages. Theoretical analysis show that the complexity of our algorithm is scalable with the dominant factor, decomposition depth. Experiments show that our algorithm has a better approximation of the k-error linear complexity than other approximation algorithms in most cases, and the simulated time complexity matches well with our theoretical result.**

1. Introduction

A sequence s is called *periodic* if there is a positive integer N such that $s_{i+N} = s_i$ for all $i \geq 0$. In cryptography, *linear complexity* is the shortest linear feedback shift register (LFSR) that can generate a given sequence. Given an N-periodic sequence $s = (s_0, s_1, \ldots, s_{N-1})^\infty$ over a finite field $\mathbb{F}$, its linear complexity denoted by $L(s)$, is the smallest integer L such that s satisfies the following linear recurrence

$$s_j + c_{L-1}s_{j-1} + \cdots + c_1 s_{j-L+1} + c_0 s_{j-L} = 0$$

for $j = L, L+1, \ldots$ with coefficients $c_0, c_1, \ldots, c_{L-1}$ in $\mathbb{F}$. Berlekamp-Massey algorithm [1] is an iterative algorithm computing the linear complexity of a finite sequence. By the Berlekamp-Massey algorithm one can recover a periodic sequence s by only knowing at most $2L(s)$ consecutive terms. Thus sequences of low linear complexity are vulnerable. Also there are security considerations on the stability of sequences. For an integer k, $0 \leq k \leq N$, the minimum linear complexity of those sequences with not more than k term changes in a period from the original sequence s is called the *k-error linear complexity* of s, denoted as $L_{N,k}(s)$, i.e.,

$$L_{N,k}(s) = \min_{w_H(e) \leq k} \{L(s+e)\},$$

where e is an N-periodic sequence over a field $\mathbb{F}$, $w_H(e)$ is the Hamming weight of e in one period, and the addition '$+$' for two sequences is defined elementwise in $\mathbb{F}$. e is also called the *error sequence* or *error pattern*.

In cryptographical applications we need to determine the stability of a periodic sequence by computing its k-error linear complexity. Efficient algorithms for computing k-error linear complexity only exist for sequences of special period over specific $\mathbb{F}$. For example, Games-Chan algorithm [2], the algorithm by Stamp and Martin [3], the algorithm by Ding, Xiao, and Shan [4], the algorithm by Kaida, Uehara and Imamura [5], and the algorithm by Kaida [6] etc. Also see the related work on the k-error linear complexity by Ming Su et al. in [7], [8], [9], [10].

However, for a periodic sequence with general period the direct approach is not feasible because the search space of all possible errors is $\sum_{t=0}^{k} \binom{N}{t}$. Considering the computational complexity of Berlekamp-Massey algorithm for computing the linear complexity is $\mathcal{O}(N^2)$, this scheme is not practical because those sequences are usually with a large period N and the number of errors k needs to be considered. So many researchers designed algorithms to approximate the k-error linear complexity of a periodic sequence.

Alecu and Sălăgean [11] proposed the modified Berlekamp-Massey algorithm (*MBM*) for approximating the k-error linear complexity, which is a search algorithm combined with heuristic method. This algorithm reduced the search space by only exploring some of all the possible error patterns, and errors were introduced only at the positions where the linear complexity is increased in the steps of the Berlekamp-Massey algorithm. Genetic algorithm (*GA*) is a probabilistic algorithm inspired by the process of natural selection, which maintains a population of candidate solutions called chromosomes, then evolves them towards better solutions using genetic operators such as selection, mutation and crossover. Alecu and Sălăgean [12] designed a genetic algorithm to approximate the k-error linear complexity of a sequence. By tuning various parameters and schemes, they showed that genetic algorithm can have a good approximation of the k-error linear complexity of a sequence. Sălăgean and Alecu [13] transformed the computation of the k-error linear complexity of a sequence to an optimization problem in the DFT domain of the sequence by Blahut's theorem. Then they gave an approximation algorithm of quadratic computational complexity by restricting the search space to error sequences whose DFT have period up to k. However,

2324-9013/16 $31.00 © 2016 IEEE
DOI 10.1109/TrustCom.2016.102

these error sequences were in extension field, and they improved this algorithm and obtained binary error sequences later [14].

The time complexity of the modified Berlekamp-Massey algorithm is still exponential, and the genetic algorithm costs a lot of storage since each population needs to be stored. The approximation algorithm using DFT gives a very rough approximation of the k-error linear complexity and the average decrease percentage of the linear complexity is not comparable with other known approximation algorithms. Therefore, we design a more efficient algorithm called the *Error Decomposition Algorithm* (*ED*) for approximating the k-error linear complexity. By decomposing the total k errors into errors of smaller weight at different stages, the search space of errors can be greatly reduced. Accordingly, the time complexity of the error decomposition algorithm is $\Theta(h\bar{c}^k N^d)$, where h denotes the number of cyclotomic cosets modulo N, d is the decomposition depth, and $\bar{c}$ is some constant between $\frac{1+\sqrt{5}}{2}$ and $\sqrt{d}+\frac{1}{2}$. This implies our algorithm is scalable to the time complexity and the accuracy of approximation with appropriate d. Particularly for $d = 3$, experiments show that our algorithm has a better approximation of the k-error linear complexity than modified Berlekamp-Massey algorithm or genetic algorithm does in most random cases. Additionally, the running time of our algorithm matches well with theoretical analysis.

The rest of paper is organized as follows. Basic definitions as well as lemmas are provided in Section 2. Then we give the error decomposition algorithm for approximating the k-error linear complexity including theoretical analysis in Section 3. Afterwards, we provide experiments including comparison of average decrease of linear complexity for different algorithms, and that for different decomposition depth of the error decomposition algorithm in Section 4. Finally we conclude in Section 5.

2. Background

First we introduce a useful notion of *k-weight complexity* as follows.

Definition 1. *Given an N-periodic sequence s over a field $\mathbb{F}$ and an integer k, $0 \le k \le N$, the k-weight complexity of s is defined as*

$$W_{N,k}(s) = \min\{L(s+e) | e \in \mathbb{F}^N, w_H(e) = k\}$$

Therefore, we have $L_{N,k}(s) = \min_{0 \le l \le k} W_{N,l}(s)$, and the following property is straightforward.

Property 1. *Given an N-periodic sequence s over a field $\mathbb{F}$, we have $L_{N,i}(s) \ge L_{N,j}(s)$ for all $i \le j$.*

The k-error linear complexity of a sequence cannot be larger than its linear complexity, however, the k-weight complexity may be larger than its linear complexity.

Next we will introduce some known results about the DFT of periodic sequences.

Definition 2. *Let s be an N-periodic sequence over $\mathbb{F}$, $\mathbb{K}$ be an extension field of $\mathbb{F}$ containing a primitive N-th root of unity α, and the DFT of s be $S = DFT(s) = (S_0, S_1, ..., S_{N-1})$. Then we have*

$$S_i = \sum_{j=0}^{N-1} s_j \alpha^{ij}, \qquad \text{for all } i = 0, 1, ..., N-1.$$

The discrete Fourier transform is an invertible, linear transformation, and the linear complexity of a periodic sequence can also be interpreted in the following way.

Blahut's Theorem. *Let s be an N-periodic sequence over $\mathbb{F}$ of characteristic p, where $\gcd(N, p) = 1$, then the linear complexity of s equals the Hamming weight of $DFT(s)$.*

Using Blahut's theorem, we can transform the optimization problem of finding the k-error linear complexity into an optimization problem in the DFT domain.

Lemma 1. *[13] Let s, e be sequences of period N over $\mathbb{F}$, $\mathbb{K}$ be an extension field of $\mathbb{F}$ that contains a primitive N-th root of unity α, $S = DFT(s)$, $E = DFT(e)$, the optimization problem of finding e which minimizes $L(s+e)$, with $w_H(e) \le k$, is equivalent to the optimization problem of finding E which minimizes $w_H(S+E)$, with $L(E) \le k$.*

Cyclotomic cosets in $\mathbb{F}_q$ will always be considered relative to powers of the cardinality q in the following part.

Lemma 2. *[15] Suppose $\gcd(N, p) = 1$, for an integer j, $0 \le j \le N-1$, let the integer k, $0 \le k \le N-1$, be an element of the cyclotomic coset C_j of j modulo N, i.e., $k \equiv jq^r \mod N$ for some integer $r \ge 0$, then we have $S_k = S_j^{q^r}$.*

By Lemma 2, elements of S whose subscripts are in the same cyclotomic coset are either all zero or all non-zero, which enables us to compute the Hamming weight of S identical to $L(s)$ efficiently.

3. Error decomposition algorithm for approximating the k-error linear complexity

Now we give an approximation algorithm for the k-error linear complexity. Rather than taking the total k errors as a whole, this algorithm takes them as a series of errors of smaller weight. Given a sequence, first we compute its k_1-weight complexity. If a decrease of the linear complexity is obtained, then we compute the k_2-weight complexity on that changed sequence, otherwise we compute the k_2-weight complexity of the original sequence. Continue until we run out of all k errors, and update the global best value if necessary. We call this process *error decomposition*. Because we do not know the best choice of $k_1, k_2, \ldots$, we have to exhaustively search. We limit $k_1, k_2, \ldots$ to be positive integers not larger than d and call d the *decomposition depth*. For simplicity we will set $d = 3$ in the following part.

The search process effectively forms a search tree. Any path from the root to a leaf has a total weight of k. Since the weight of intermediate steps may overlap with each other, the sequence corresponding to the leaf node cannot have a

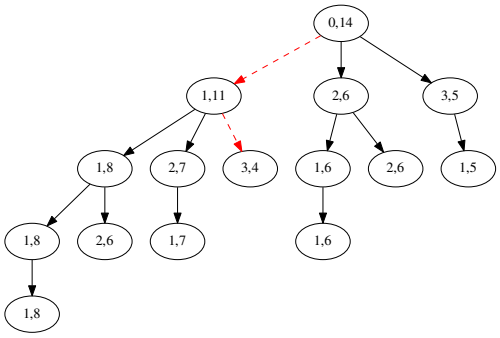


Figure 1: Example of the Error Decomposition Algorithm, Tree of Recursive Calls for the Sequence s = 010000011101001, k = 4

number of errors larger than k. At each path from the root to a leaf, the linear complexity at each step decreases or stays the same, thus we expect it to give a good approximation of the k-error linear complexity. Note that at each step of computing the k_1-weight complexity, even if no decrease is obtained, we subtract k_1 from the remaining allowed errors to avoid infinite loop.

We illustrate our algorithm with an example.

Example 1. *Applying the error decomposition algorithm to* $s = 010000011101001$ *(*$N = 15$*), whose linear complexity is* 14*,* $k = 4$*. The corresponding error sequence is* 100001010000010*.*

Figure 1 shows the tree of recursive calls for the error decomposition algorithm. Each node contains two numbers, the first number k_i means that the k_i-weight complexity is being computed at the current step, and the second is the resulting linear complexity of the new sequence. The leaf nodes have the final result of each path in the tree. In our example, the optimal path is drawn in red dashed line, i.e., we first compute the 1-weight complexity of s, which results in a new sequence s_1 of linear complexity 11, then we compute the 3-weight complexity of s_1, and get another new sequence of linear complexity 4, which is the best result found by our algorithm. The corresponding error sequence is 100001010000010.

We denote by $f(N,k)$ capacity of the search space explored by Algorithm 2. By studying the structure of the tree of recursive calls, we have a recurrence relation of $f(N,k)$ for $k > 3$ as follows:

$$f(N,k) = f(N,k-3) + f(N,k-2) + f(N,k-1) + \binom{N}{1} + \binom{N}{2} + \binom{N}{3}, \quad (1)$$

Algorithm 1 The WEIGHT-COMPLEXITY Subroutine

Input: s a sequence of period N over field $\mathbb{F}$, N not divisible by the characteristic of $\mathbb{F}$, $k \leq N$
Output: a sequence ss such that $L(ss)$ equals the minimum of $L(s)$ and $W_{N,k}(s)$

```
Compute the cyclotomic coset leaders modulo N, c_1, c_2, ..., c_h, and the cardinality of each coset, l_1, l_2, ..., l_h
ss_best = s, sdft = DFT(s), best = w_H(sdft)
for each (p_1, p_2, ..., p_k) tuple of error position do
    tmp = sdft
    for each i ∈ {c_1, c_2, ..., c_h} do
        E_i = α^{ip_1} + α^{ip_2} + ··· + α^{ip_k}
        if S_i ≠ 0 and E_i == S_i then
            tmp = tmp − l_i
        else if S_i == 0 and E_i ≠ S_i then
            tmp = tmp + l_i
        end if
    end for
    if tmp < best then
        Update ss_best, best
    end if
end for
return ss_best
```

with

$$f(N,1) = \binom{N}{1}$$
$$f(N,2) = f(N,1) + \binom{N}{1} + \binom{N}{2}$$
$$f(N,3) = f(N,1) + f(N,2) + \binom{N}{1} + \binom{N}{2} + \binom{N}{3}$$

as initial value.

We are interested in the order of magnitude of $f(N,k)$. Solving Equation (1) would require working with complex field, however, with simple amplification and minification of $f(N,k)$, we are able to obtain a lower bound and an upper bound, which give us enough insight into the order of magnitude of $f(N,k)$.

Theorem 1. *For an* N*-periodic sequence with the number of errors* k*, capacity of the search space of the error decomposition algorithm is* $\Theta(c^k N^3)$*, where* c *is a constant between* $\frac{1+\sqrt{5}}{2}$ *and* 2*.*

Proof. $f(N,k)$ is a positive monotonically increasing function of k. Let $m = \binom{N}{1} + \binom{N}{2} + \binom{N}{3}$, we have

$$f(N,k) < f(N,k-1) + 2f(N,k-2) + m, \quad (2)$$

and

$$f(N,k) > f(N,k-1) + f(N,k-2) + m. \quad (3)$$

Algorithm 2 The Error Decomposition Algorithm

Input: s a sequence of period N over field $\mathbb{F}$, N not divisible by the characteristic of $\mathbb{F}$, $k \leq N$
Output: $BEST$ the approximate k-error linear complexity of s and the corresponding error sequence

```
BEST = L(s)
SBEST = s
procedure SEARCH(s, k)
    if k == 0 and L(s) < BEST then
        BEST = L(s)
        SBEST = s
    end if
    if k ≥ 1 then
        ss = WEIGHT-COMPLEXITY(s, 1)
        SEARCH(ss, k − 1)
    end if
    if k ≥ 2 then
        ss = WEIGHT-COMPLEXITY(s, 2)
        SEARCH(ss, k − 2)
    end if
    if k ≥ 3 then
        ss = WEIGHT-COMPLEXITY(s, 3)
        SEARCH(ss, k − 3)
    end if
end procedure
SEARCH(s, k)
return BEST, SBEST − s
```

From (2) we derive

$$\begin{aligned} f(N,k) < \frac{1}{3}(2^{k-1} - 2 - \frac{(-1)^{k-2}-1}{2})m \\ + \frac{1}{3}(2^{k-1} - (-1)^{k-1})f(N,2) \\ + \frac{1}{3}(2^{k-1} + 2(-1)^{k-1})f(N,1). \end{aligned} \tag{4}$$

From (3) we derive

$$\begin{aligned} f(N,k) > \frac{1}{\sqrt{5}}(\varphi^k - (-\varphi^{-1})^k - \varphi^2 + \varphi^{-2})m \\ + \frac{1}{\sqrt{5}}(\varphi^{k-1} - (-\varphi^{-1})^{k-1})f(N,2) \\ + \frac{1}{\sqrt{5}}(\varphi^{k-2} - (-\varphi^{-1})^{k-2})f(N,1), \end{aligned} \tag{5}$$

where $\varphi = \frac{1+\sqrt{5}}{2}$ and $-\varphi^{-1} = \frac{1-\sqrt{5}}{2}$.

In both (4) and (5), the first term of the right side dominates. Since m is of order $\Theta(N^3)$, the conclusion holds. □

Analogously, for general decomposition depth d we have

$$f(N,k) < f(N,k-1) + (d-1)f(N,k-2) + \Theta(N^d),$$

and

$$f(N,k) > f(N,k-1) + f(N,k-2) + \Theta(N^d).$$

Thus, we obtain an upper bound and a lower bound of $f(N,k)$ as follows:

$$f(N,k) = \mathcal{O}\Big((\frac{1+\sqrt{1+4(d-1)}}{2})^k N^d\Big) = \mathcal{O}\Big((\sqrt{d}+\frac{1}{2})^k N^d\Big),$$

and

$$f(N,k) = \Omega\Big((\frac{1+\sqrt{5}}{2})^k N^d\Big).$$

For each error pattern e in the search space explored by the error decomposition algorithm, we have to compute the Hamming weight of $DFT(s+e)$. Since we calculate it in a cyclotomic coset basis in Algorithm 1 and $DFT(s)$ is computed beforehand, the time complexity of computing the Hamming weight of $DFT(s+e)$ is $\Theta(h)$, where h denotes the number of cyclotomic cosets modulo N. Combined with Theorem 1, we determine the time complexity of the error decomposition algorithm as follows.

Corollary 1. *The time complexity of the error decomposition algorithm for approximating the k-error linear complexity of a sequence of period N is $\Theta(hc^kN^3)$, where h denotes the number of cyclotomic cosets modulo N, c is some constant between $\frac{1+\sqrt{5}}{2}$ and 2. For general d, the time complexity is $\Theta(h\bar{c}^kN^d)$, where $\bar{c}$ is some constant between $\frac{1+\sqrt{5}}{2}$ and $\sqrt{d}+\frac{1}{2}$.*

Next we will present the experimental results of the running time of Algorithm 2 to see how it matches with our theoretical analysis.

4. Experimental Results

We have done experiments to measure the accuracy of the error decomposition algorithm for the k-error linear complexity and compared the results with modified Berlekamp-Massey algorithm and genetic algorithm. We choose different combinations of N and k, with the total search space ranging from hundreds of thousands to tens of billions. For each (N,k) pair, we randomly generate 10 sequences using linear congruential generators and compute their approximate k-error linear complexity by the error decomposition algorithm, modified Berlekamp-Massey algorithm and the genetic algorithm. These algorithms are implemented using GAP [16] and the full experiment environment settings are listed in Table 1.

Table 2 shows the average decrease of linear complexity, i.e., $(L(s) - result)/L(s)$, where $result$ is the approximate k-error linear complexity each algorithm returns.

From Table 2 we can see that the error decomposition algorithm is able to compute a good upper bound of the k-error linear complexity. In most cases, the decrease of the linear complexity obtained by our algorithm is better than that of the modified Berlekamp-Massey algorithm and the genetic algorithm.

TABLE 1: The Experiment Environment

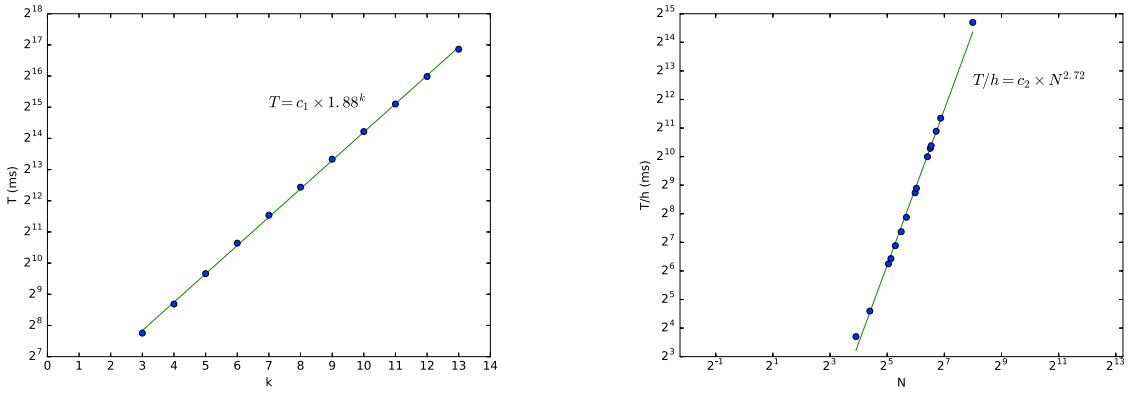
CPU	Memory	OS	GAP
Intel Core i3-2310M @ 2.10GHz	4GB	Linux 4.4	4.7.9

TABLE 2: Comparison of Average Decrease Percentage Between Approximation Algorithms

Algs \ (N, k)	(33, 5)	(45, 5)	(51, 6)	(63, 7)	(65, 8)	(63, 9)	(85, 8)	(91, 9)	(93, 9)
MBM	25.0	31.3	30.7	35.8	26.8	38.1	30.4	27.9	31.4
GA	36.4	37.8	33.7	32.0	22.3	32.9	21.9	17.3	20.9
ED	35.7	41.2	41.3	39.2	28.0	43.3	31.1	23.5	30.8

TABLE 3: Comparison of Average Decrease Percentage Between Different Decomposition Depths

depth \ (N, k)	(33, 5)	(45, 5)	(51, 6)	(63, 7)	(65, 8)	(63, 9)	(85, 8)	(91, 9)	(93, 9)
2	33.2	27.5	25.0	32.6	19.6	29.3	23.5	18.5	19.4
3	36.7	38.0	39.3	43.8	25.6	40.7	29.9	27.2	31.6
4	41.7	42.4	43.5	47.6	33.1	50.8	35.5	32.3	35.6



(a) let $N = 63$ and increase k

(b) let $k = 8$ and increase N

Figure 2: The Running Time of Our Algorithm with the Increase of N and k

We calculated the ratio of the total search space over the search space of our algorithm. For our chosen (N, k) pair listed in Table 2, the ratio goes up to 100000 when the total search space ranges from hundreds of thousands to tens of billions, which is a considerable save of the search space, and gives the reason why our heuristic method is effective. Also, the error decomposition process is less likely to fall into local optimum than the heuristic method of the modified Berlekamp-Massey algorithm.

Our algorithm is memory efficient because the search space is constructed implicitly by the search tree, which consumes $\mathcal{O}(k)$ stack space, while in the case of genetic algorithm each population needs to be stored explicitly in memory.

We also have observed how the running time of the error decomposition algorithm is increased with the increase of N and k respectively. Figure 2a shows how the running time is increased when we let $N = 63$ and increase k from 3 to 12. Figure 2b shows how the running time is increased when we let $k = 8$ and increase N from 15 to 255. $T(k)$ and $T(N)$ denote the running time related to parameters k and N respectively, h denotes the number of cyclotomic cosets modulo N and can be precomputed. Figure 2a shows that the running time is proportional to 1.88^k, implying the running time roughly doubles when we increase k by 1, and Figure 2b shows that the running time is proportional to $hN^{2.72}$. Combining them we get a running time of $\mathcal{O}(h1.88^k N^{2.72})$, which matches well with our theoretical analysis. Moreover, the error decomposition algorithm is scalable. Depending on how much computing resources we have and what accuracy we want, we can set the decomposition depth d to be appropriate.

In order to compare different decomposition depths, we list the average decrease of linear complexity obtained by depth of 2, 3 and 4 for each N, k pair in Table 3. Because the search space of error decomposition of depth d_1 is a subset of that with depth d_2 if $d_1 < d_2$, error decomposition of depth d_2 will consume more computing resource and guarantee a better approximation result than error decomposition of depth d_1. Note that when the depth is set to 4, the error decomposition algorithm already has a distinct advantage in terms of accuracy compared with the modified Berlekamp-Massey algorithm and the genetic algorithm.

5. Conclusion

We have proposed a scalable error decomposition algorithm for approximating the k-error linear complexity of periodic sequences. For small k, we use Blahut's theorem and

the cyclotomic structure of DFT to compute the k-weight complexity. Then by decomposing large k-errors into errors of smaller weight at different stages, we can quickly reduce the search space. The proposed approximation algorithm is efficient and has time complexity $\Theta(hc^kN^3)$. We have implemented this algorithm and compared it with modified Berlekamp-Massey algorithm and the genetic algorithm. The experiments show that our algorithm gives a better approximation of the k-error linear complexity than other algorithms in most random cases. Moreover, the simulated running time matches well with our theoretical analysis. Also, we generalize this algorithm to different decomposition depth d, and the time complexity of corresponding algorithm is $\Theta(h\bar{c}^kN^d)$, in which the decomposition depth d dominates and the number of errors k is negligible. As future work, we will consider non-binary case as well as exploring larger search space with lower time complexity. Note that the proposed method can be extended to study stability of other complexity measures of sequences.

Acknowledgments

This work is partially supported by NSF of China (grant numbers: 61373018, 11301288, 11550110491, 61003070), Program for New Century Excellent Talents in University (grant number: NCET130301) and the Fundamental Research Funds for the Central Universities (grant number: 65141021). The second corresponding author is also supported by China Scholarship Council.

References

[1] J. L. Massey, "Shift-register synthesis and bch decoding," *Information Theory, IEEE Transactions on*, vol. 15, no. 1, pp. 122–127, 1969.

[2] R. A. Games and A. H. Chan, "A fast algorithm for determining the complexity of a binary sequence with period 2^n," *Information Theory, IEEE Transactions on*, vol. 29, no. 1, pp. 144–146, 1983.

[3] M. Stamp and C. F. Martin, "An algorithm for the k-error linear complexity of binary sequences with period 2^n," *Information Theory, IEEE Transactions on*, vol. 39, no. 4, pp. 1398–1401, 1993.

[4] C. Ding, G. Xiao, and W. Shan, *The stability theory of stream ciphers*. Springer Science & Business Media, 1991, vol. 561.

[5] T. Kaida, S. Uehara, and K. Imamura, "An algorithm for the k-error linear complexity of sequences over gf(p^m) with period p^n, p a prime," *Information and Computation*, vol. 151, no. 1, pp. 134–147, 1999.

[6] T. Kaida, "On the generalized lauder-paterson algorithm and profiles of the k-error linear complexity for exponent periodic sequences," in *Sequences and Their Applications-SETA 2004*. Springer, 2004, pp. 166–178.

[7] M. Su, "Decomposing approach for error vectors of k-error linear complexity of certain periodic sequences," *IEICE Transactions*, vol. 97-A, no. 7, pp. 1542–1555, 2014.

[8] F. Fu, H. Niederreiter, and M. Su, "The characterization of 2^n-periodic binary sequences with fixed 1-error linear complexity," in *Sequences and Their Applications - SETA 2006, 4th International Conference, Beijing, China, September 24-28, 2006, Proceedings*, 2006, pp. 88–103.

[9] M. Su and L. Lu, "The properties of the 1-error linear complexity of p^n-periodic sequences over $\mathbb{F}_p$," in *IEEE International Symposium on Information Theory 2006, Seattle, USA, July 9-14, 2006*. IEEE, 2006, pp. 1998–2002.

[10] M. Su, "Decomposing approach for error vectors of k-error linear complexity of 2^n-periodic binary sequences," in *WCC(Workshop on Coding and Cryptography) preproceeding 2009, Ullensevang, Norway, May, 2009, preproceedings*, 2009, pp. 399–415.

[11] A. Alecu and A. Sălăgean, "Modified berlekamp-massey algorithm for approximating the k-error linear complexity of binary sequences," in *Cryptography and Coding*. Springer, 2007, pp. 220–232.

[12] A. Alecu and A. Sălăgean, "A genetic algorithm for computing the k-error linear complexity of cryptographic sequences," in *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*. IEEE, 2007, pp. 3569–3576.

[13] A. Alecu and A. Sălăgean, "An approximation algorithm for computing the k-error linear complexity of sequences using the discrete fourier transform," in *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*. IEEE, 2008, pp. 2414–2418.

[14] A. Sălăgean and A. Alecu, "An improved approximation algorithm for computing the k-error linear complexity of sequences using the discrete fourier transform," in *Sequences and Their Applications–SETA 2010*. Springer, 2010, pp. 151–165.

[15] W. Meidl and H. Niederreiter, "On the expected value of the linear complexity and the k-error linear complexity of periodic sequences," *Information Theory, IEEE Transactions on*, vol. 48, no. 11, pp. 2817–2825, 2002.

[16] *GAP – Groups, Algorithms, and Programming, Version 4.7.9*, The GAP Group, 2016. [Online]. Available: http://www.gap-system.org