Conditional Cube Searching and Applications on Trivium-Variant Ciphers

Xiaojuan Zhang[1,2](✉), Meicheng Liu[1,2], and Dongdai Lin[1,2]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
zhangxiaojuan@iie.ac.cn

[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Abstract. In this paper, we describe a new cube searching method called conditional searching. The main idea of this new searching method is to reduce the searching space and contains two main steps: finding complementary variables and searching conditional cubes. At the first step, we introduce a concept of complementary variables corresponding to cube variables to ensure that cube variables are not multiplied with each other in the first few propagations. According to the taps in the feedback functions, two main strategies are given to find complementary variables. At the second step, we first give a simple algorithm to estimate the maximal size of conditional cubes that don't contain any complementary variable. Then another algorithm is given to search conditional cubes. We can confirm the maximum numbers of initialization rounds of some NFSR-based cryptosystems such that the generated keystream bit does not achieve the maximum algebraic degree with our cube searching method and the algebraic degree estimated method numeric mapping. We apply our method to Trivium to verify the validity and our searching space is about $2^{12.5}$ much smaller than that of existing results. We also introduce two Trivium-variants named Par-Trivium and Loc-Trivium, and apply the method to them. We can get an upper bound of the maximum initialization rounds when we change the parameters or the key and IV loading locations in Trivium. The applications provide some insights into the taps used in the feedback functions of such stream ciphers. We believe that our method is useful in both cryptanalysis and design of NFSR-based cryptosystems.

Keywords: Cryptanalysis · Numeric mapping · Stream cipher Trivium · Trivium variants

1 Introduction

A nonlinear feedback shift register (NFSR) is widely used in modern cryptographic primitives, especially in radio-frequency identification devices (RFID) and wireless sensor networks applications. Most NFSR-based cryptosystems can be described as tweakable Boolean functions with respect to both secret variables (e.g., key bits) and public variables (e.g., plaintext bits or initial value (IV) bits). The algebraic degrees of these Boolean functions are of great importance in the security of the corresponding primitives. For a cryptographic primitive with low algebraic degree, it is vulnerable to many known attacks, such as cube attacks [1–4] and higher order differential attacks [5,6] which are the most powerful cryptanalytic tools against NFSR-based cryptosystems. Cube attack is introduced by Dinur and Shamir [1], which is a chosen plaintext key-recovery attack. Since then, cube attack has attracted much attention in recent public cryptographic literatures. At Eurocrypt 2015, Dinur et al. [2] publish a key-recovery attack on Keccak keyed modes, where the cube variables are selected not to multiply with each other after the first round, then the output degree of the polynomials is reduced. Later Huang et al. [7] propose a new conditional cube attack on Keccak keyed modes and present an 8-round attack on Keyak. Recently, conditional cube attack is applied to round-reduced ASCON [8] and River Keyak [9]. Also, algebraic attacks [10,11] and integral attacks [12] are easy to perform on a cryptographic primitive with low algebraic degree.

For modern cryptographic primitives, it is difficult to compute the exact values of the algebraic degrees. But, there are several theoretical tools can be used to estimate the upper bounds on the algebraic degrees of iterated permutations, and concurrently exploited to attack iterated ciphers [1,13–15]. Yet for NFSR, there are few tools for estimating its algebraic degree, besides symbolic computation and statistical analysis. Some techniques highly depend on computational capabilities which restrict the cryptanalytic results. A variant of cube attacks called dynamic cube attacks can reach much higher attack complexity, but they are still limited by the size of the cubes [1,4]. Based on this point, in either cube attacks or cube testers, the cubes with size larger than 54 have never been utilized in cryptanalysis of NFSR-based cryptosystems. Recently, at CRYPTO 2017, two works on cube attacks use the cubes with size larger than 50. The one by Todo et al. [16] presents possible key recovery attacks against Trivium [17], Grain-128a [18] and ACORN [19] using the cubes of sizes 72, 92 and 64. They mainly make use of the propagation of the bit-based division property of stream ciphers. The other by Liu [20] gives a tool called numeric mapping to iteratively obtain the upper bounds on the algebraic degrees of NFSR-based cryptosystems.

Our Contributions. In this paper, we propose a new cube searching method named conditional searching. The main idea of this new searching method is to reduce the searching space through controlling the propagation of the IV bits and contains two main steps: finding complementary variables and searching conditional cubes. At the finding complementary variables step, we introduce a concept of complementary variables corresponding to cube variables. To ensure that cube variables are not multiplied with each other in the first few propagations, we give two main strategies to find complementary variables according to the taps used in the feedback functions and the size of cubes needed. At the second step, we first give a simple algorithm to estimate the maximal size of conditional cubes which means that there is not any corresponding complementary variable. Then another algorithm is given to search conditional cubes. With our cube searching method and the algebraic degree estimated method numeric mapping introduced by Liu [20], we can confirm the maximum numbers of initialization rounds of some NFSR-based cryptosystems such that the generated keystream bit does not achieve the maximum algebraic degree.

We apply our method to Trivium to verify the validity with the searching space of size about $2^{12.5}$. While in [20], the searching space is 2^{25}. Our searching space is smaller than that of the existing results. We also apply the method to Trivium-variants containing Par-Trivium and Loc-Trivium. For Par-Trivium, where the parameters in Trivium are changed, we estimate the algebraic degrees and can get an upper bound of the maximum initialization rounds such that the generated keystream bit does not achieve the maximum algebraic degree. The experiments show that the maximum round of Par-Trivium is 863 which is the worst case. So, parameters in Par-Trivium can be as big as possible on the premise of the security against other attacks. And for Loc-Trivium, where the key and IV loading locations are changed in Trivium, we can get similar results. The experiments show that the maximum initialization round of all considered Trivium-variants is 910, which is the worst case and should be avoided to be resistant to cube attacks or cube tests when new ciphers are designed. The applications provide some insights into the taps used in the feedback functions and the key and IV loading locations of such stream ciphers, which are useful in both cryptanalysis and design of NFSR-based cryptosystems.

Organization of the Paper. The rest of this paper is structured as follows. In Sect. 2, basic definitions and notations are provided. Section 3 shows the general framework of our conditional searching method, while its applications on Trivium and Trivium-variants are given in Sects. 4 and 5. Section 6 concludes the paper.

2 Preliminaries

Boolean Functions and Algebraic Degree. Let $\mathbb{F}_2$ be the binary field and $\mathbb{F}_2^n$ the n-dimensional vector space over $\mathbb{F}_2$. An n-variable Boolean function is a mapping from $\mathbb{F}_2^n$ into $\mathbb{F}_2$. An n-variable Boolean function f can be uniquely represented as a multivariate polynomial over $\mathbb{F}_2$,

$$f(x_1, x_2, \cdots x_n) = \bigoplus_{c=(c_1,c_2,\cdots c_n)\in\mathbb{F}_2} a_c \prod_{i=1}^{n} x_i^{c_i}, \quad a_c \in \mathbb{F}_2,$$

called the algebraic normal form (ANF). The algebraic degree of f, denoted by $\deg(f)$, is defined as $\max\{wt(c)|a_c \neq 0\}$, where $wt(c)$ is the Hamming weight of c.

Numeric Mapping. Let $f(x_1, x_2, \cdots x_n) = \bigoplus_{c=(c_1,c_2,\cdots c_n)\in\mathbb{F}_2} a_c \prod_{i=1}^{n} x_i^{c_i} (a_c \in \mathbb{F}_2)$ be a Boolean function. Denote by $\mathbb{B}_n$ the set of all n-variable Boolean functions. The numeric mapping [20], denoted by $\mathtt{DEG}$ is defined as

$$\begin{aligned} \mathtt{DEG}: \quad & \mathbb{B}_n \times \mathbb{Z}_n \to \mathbb{Z}, \\ & (f, D) \mapsto \max_{a_c \neq 0} \{\sum_{i=1}^{n} c_i d_i\}, \end{aligned} \tag{1}$$

where $D = (d_1, d_2, \cdots, d_n)$ and a_c's are coefficients of the ANF of f. Let $g_i (1 \leq i \leq n)$ be Boolean functions on m variables, and denote $\deg(G) = (\deg(g_1), \deg(g_2), \cdots, \deg(g_n))$ for $G = (g_1, g_2, \cdots, g_n)$. The numeric degree of the composite function $h = f \circ G$ is defined as $\mathtt{DEG}(f, \deg(G))$, denoted by $\mathtt{DEG}(h)$ for short. The algebraic degree of h is always less than or equal to the numeric degree of h. The algebraic degrees of the output bits with respect to the internal states can be estimated iteratively for NFSR-based cryptosystems by using numeric mapping.

Cube Testers. Given a Boolean function f and a term t_I containing variables from an index subset I that are multiplied together, the function can be written as

$$f(x_1, x_2, \cdots, x_n) = f_S(I) \cdot t_I \oplus q(x_1, x_2, \cdots, x_n),$$

where the terms in $q(x_1, x_2, \cdots, x_n)$ miss at least one variable from I and $f_S(I)$ is called the superpoly of I in f. The basic idea of cube testers is that

$$\sum_{x' \in C_I} f = f_S(I),$$

where C_I are all possible values of the subset of variables in the term t_I. The target of cube testers work by evaluating superpolys of carefully selected terms t_I's which are products of public variables (e.g., IV bits), and trying to distinguish them from a random function. A cube tester can detect the nonrandomness in cryptographic primitives by extracting the testable properties of the superpoly, such as unbalance, constantness and low degree, with the help of property testers. Especially, the superpoly $f_S(I)$ is equal to a zero constant, if the algebraic degree of f in the variables from I is smaller than the size of I.

3 A New Method for Searching Cube

In this section, we propose a new model for searching cube, called conditional searching. The new searching method consists of two phases, finding complementary variables and searching conditional cubes. First, we will give a generalized model of the initialization phases of NFSR-based cryptosystems.

3.1 Generalized Model

For NFSR-based cryptosystems, especially NFSR-based stream ciphers, the initialization phase is used to initialize the internal state using secret variables (e.g., key bits) and public variables (e.g., plaintext bits or IV bits). Then the encryption phase just consists of an exclusive or (XOR) with the continuously updated keystream. The generalized model of initialization phases of some NFSR-based cryptosystems can be depicted as Fig. 1, which is helpful in the sense that we could study some special properties/choices more clearly in a unified framework.

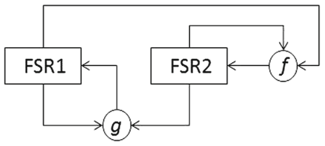

Fig. 1. Generalized initialization phase of NFSR-based cryptosystem

FSR1 and FSR2 are two registers. Here we stress that FSR1 in the model can be further decomposed into a series of cascaded smaller NFSRs or LFSRs, which could also be treated by our cryptanalysis. There are two Boolean functions involved in the model: a (either linear or non-linear) Boolean function g and a non-linear Boolean function f. FSR2 is initialized by the padded IV. It is obvious that our generalized model could cover initialization processes of Grain v1 [21], Trivium and so on. Denote by S_t and B_t the initial states of FSR1 and FSR2 at time t with size of m_1 and m_2. At each step, FSR2 is updated by f, sometimes without any taps from FSR1, and FSR1 is updated by g as follows:

- FSR1 is updated recursively by g as $S_{t+1} = (s_{t+1}, s_{t+2}, \cdots, s_{t+m_1})$ with $s_{t+m_1} = g(S_t, B_t)$.
- FSR2 is updated recursively by f as $B_{t+1} = (b_{t+1}, b_{t+2}, \cdots, b_{t+m_2})$ with $b_{t+m_2} = f(B_t, S_t)$ or $b_{t+m_2} = f(B_t)$.
- We assume these processes are invertible, and the inverse processes are $S_{t-1} = (s_{t-1}, s_t, \cdots, s_{t+m_1-2})$ with $s_{t-1} = g^{-1}(S_t, B_t)$ and $B_{t-1} = (b_{t-1}, b_t, \cdots, b_{t+m_2-2})$ with $b_{t-1} = f^{-1}(B_t, S_t)$ or $b_{t-1} = f^{-1}(B_t)$.

3.2 Conditional Searching Method

Huang et al. [7] propose a new conditional cube attack on Keccak keyed modes and present an 8-round attack on Keyak. By restraining some bit conditions of the key, they obtain a new set of cube variables which not only do not multiply with each other after the first round, but also contains one cube variable that does not multiply with other cube variables after the second round, and then the output degree over cube variables is further reduced. Based on this method, we give our conditional searching method which is a searching tool with some conditions to reduce the searching space. The conditional cubes used in our paper is different, which means that the cubes do not contain any complementary variables. Before the method is given, we will give a definition of complementary variables. In the cube attack or cube test against stream cipher, cube variables are chosen from the IV bits with length l, where $IV = (iv_1, iv_2, \cdots, iv_l)$. When iv_i is chosen as a cube variable, the variables that must not be chosen are called complementary variables corresponding to the cube variable iv_i. Our conditional searching method contains two steps: finding complementary variables and searching conditional cubes. We call the cubes that do not contain any corresponding complementary variables, conditional cubes for simplicity. The conditional cubes and the maximum numbers of initialization rounds (maximum rounds for simplicity) are corresponding to the situation that the generated keystream bit does not achieve the maximum algebraic degree.

Finding Complementary Variables. The main way to find complementary variables is to control the propagation of cube variables. As shown in the generalized model, the IV bits are loaded in FSR2 and cube variables are chosen from them. This way, the propagation paths of the IV bits are of importance to choose cube variables and determine the corresponding complementary variables. In the beginning, the IV bits only appear in FSR2 and take part in the update of FSR2 (or FSR1) through the feedback function f (or g). Several steps later, they will appear in both FSR2 and FSR1.

It is intuitive that if the cube variables are not multiplied with each other in the first few propagations, the maximum initialization round, such that the generated keystream bit does not achieve the maximum algebraic degree, will be larger. The complementary variables are determined by these variables that would be multiplied with each other. For the multiplied variables, if one of them is chosen to be cube variable, the others are defined as complementary variables. This way, the taps in the feedback functions play a leading role in finding complementary variables. It is easy to see that the more propagations are controlled, the less conditional cubes are satisfied. Two main strategies to choose cube variables are used here.

– In the beginning, where the IV bits take part in the first iteration, cube variables should not be multiplied with each other.
– When the IV bits as a part of the feedback value take part in the update function, cube variables must not be multiplied with each other.

By an iteration we mean two or more rounds which depends on the maximal tap in the feedback function, more details, one can see the following example.

Example 1. Let $m_1 = m_2 = 8$ and

$$\begin{aligned} s_{t+8} &= s_{t+6}s_{t+1} + s_t + b_{t+2}, \\ b_{t+8} &= b_{t+4}b_{t+3} + b_t + s_{t+3}, \\ S_1 = (s_1, s_2, &\cdots, s_8) = (iv_1, iv_2, \cdots, iv_8), \end{aligned}$$

where $t \geq 1$. For FSR2, the maximal tap in the feedback function is s_{t+6} and an iteration means two rounds, where

$$\begin{array}{r l}
\text{the first iteration} & \begin{cases} \mathbf{s_9} = s_7 s_2 + s_1 + b_3 \\ s_{10} = s_8 s_3 + s_2 + b_4 \end{cases} \\
\text{the second iteration} & \begin{cases} \mathbf{s_{11}} = \mathbf{s_9} s_4 + s_3 + b_5 \\ s_{12} = s_{10} s_5 + s_4 + b_6 \end{cases} \\
\text{the third iteration} & \begin{cases} s_{13} = \mathbf{s_{11}} s_6 + s_5 + b_7 \\ \cdots \end{cases}
\end{array}$$

What we need to control is the first two iterations, that is $t \leq 4$. In the first iteration, s_7, s_2 and s_8, s_3 are multiplied with each other. In the second iteration, s_9 is multiplied with s_4 and s_{10} is multiplied with s_5, where s_9 and s_{10} are feedback values. Take the IV bits into account, the multiplied pairs are $(iv_2, iv_7), (iv_3, iv_8), (iv_1, iv_4)$ and (iv_2, iv_5), during the first two iterations. We can predict that if iv_i is a cube variable, iv_{i+3} and iv_{i+5} will be multiplied with iv_i at some point. So, the complementary variables are iv_{i+3} and iv_{i+5} corresponding to the cube variable iv_i. Similarly, for FSR1, when $t = 9$, the number of the iterations with respect to S_1 is larger than two. The multiplied pairs are $(iv_4, iv_5), (iv_5, iv_6)$ and (iv_6, iv_7), during the first two iterations and the complementary cube variable is iv_{i+1} corresponding to the cube variable iv_i. In summary, if iv_i is chosen to be a cube variable, the set of complementary variables is $\{iv_{i+1}, iv_{i+3}, iv_{i+5}\}$.

Searching Conditional Cube. Once the complementary variables are obtained, we need to search the conditional cubes that do not contain any complementary variables. But before that, we have to determine the maximal size of conditional cubes and then to search this kind of cubes. If the number of variables that can be used as cube variables is small, the problem is very easy. As shown in Example 1, the maximal size of conditional cubes is four, which are $\{iv_1, iv_3, iv_5, iv_7\}$ and $\{iv_2, iv_4, iv_6, iv_8\}$. We can verify that among conditional cubes, cube variables will not be multiplied with each other in the first few rounds and there is not any complementary variables $\{iv_{i+1}, iv_{i+3}, iv_{i+5}\}$ corresponding to iv_i.

In the NFSR-based stream ciphers, the length of the IV bits is so large that we can't easily estimate the maximal size and search the cubes. Two algorithms are given to solve these two problems. Let the set of the complementary variables corresponding to iv_i be $C' = \{iv_{i+j_1}, iv_{i+j_2}, \cdots, iv_{i+j_m}\}$. Algorithm 1 is used to estimate the maximal size of conditional cubes. The main idea is to choose one special cube according to the indexes of the IV bits from small to large. It is obvious that the size *dim* of this special cube is the maximal size and all the conditional cubes are of size less than *dim*.

Algorithm 2 is given to search conditional cubes with the maximal size *dim*. First, we evenly divide the IV bits into several parts. For the first part, search all sub-conditional cubes with possible sizes. For other parts, we can get the corresponding sub-conditional cubes by changing the subscripts, as show in line 7 of

Algorithm 1. Estimation of maximal size

```
Require: the complementary variables set $C^{'}$, the set $C = \phi$ , and the length $l$ of the IV bits
Ensure: the maximal size of conditional cubes
1: for $i$ from 1 to $l$ do
2:     if $iv_i \notin C^{'}$ then
3:         Add $iv_i$ to $C$.
4:         Add $iv_{i+j_1}, iv_{i+j_2}, \cdots, iv_{i+j_m}$ to $C'$.
5:     end if
6: end for
7: Let $dim = |C|$, where $|C|$ is the size of set $C$.
8: return  $dim$ is the maximal size of conditional cubes.
```

Algorithm 2. Searching conditional cubes

```
Require: the maximal size $dim$, the complementary variables set $C^{'}$, $l$ , $m$
Ensure: the maximal round of conditional cubes
1: for $i$ from 1 to $m$ do
2:     In the set $\{iv_1, iv_2, \cdots, iv_m\}$, search all possible sub-conditional cubes with size
       $i$ and denoted by $C_i$. Denote the size of $C_i$ by $SC_i$.
3: end for
4: for all combinations of sub-conditional cubes with size $subdim_j$, such that
   $subdim_1 + subdim_2 + \cdots + subdim_{l/m} = dim$ do
5:     for $j$ from 1 to $l/m$ do
6:         Choose one sub-conditional cube in $C_{subdim_j}$
7:         Add $(j-1) \cdot m$ to the subscripts of the sub-conditional cube.
8:     end for
9:     Put the sub-conditional cubes with size $subdim_j$ together to obtain a cube with
       size $dim$
10:    Test whether the cube with size $dim$ is a conditional cube.
11:    if the cube is an conditional cube then
12:        Estimate the maximum round with numeric mapping method.
13:    end if
14: end for
15: return  the maximal round of conditional cubes
```

Algorithm 2. Second, examine the combinations of all possible sub-conditional cubes to obtain the conditional cubes. Denote the length of the IV bits and the evenly divided factor by l and m. Evenly divide the IV bits into l/m parts, denoted by part j, where $1 \leq j \leq l/m$. Denoted by $subdim_j$ the sub-size of sub-conditional cubes chosen from part j. The criteria of testing whether the cube with size dim is a conditional cube is to verify that if iv_i is a cube variable, whether the complementary variables in $C^{'}$ are cube variables. For example, if the length of the IV bits is 80, we can evenly divide them into 4 parts, iv_1 to iv_{20}, iv_{21} to iv_{40}, iv_{41} to iv_{60}, and iv_{61} to iv_{80}. The first part is iv_1 to iv_{20} and all sub-conditional cubes with possible sizes can be obtained by the simple exhaustive search method. For other parts, add 20, 40, 60 to the subscripts

of the sub-conditional cubes. If the size of conditional cubes is 38, we need to consider all possible combinations of sub-conditional cubes, where the sum of the sub-sizes is 38. Then for each cube variable iv_i, verify whether the corresponding complementary variables are cube variables. If not, this cube is a conditional cube.

The main time complexity of Algorithm 2 is to test the combinations of sub-conditional cube with size of $subdim_j$, where $1 \leq j \leq l/m$. According to the length of the IV bits and the maximum size of conditional cubes, m need to be chosen carefully, which plays an important role in the time complexity. Whether the IV bits need to be divided depends on the complementary variables. For Trivium, two complementary variables are iv_{i+1} and iv_{i-1} and we can exhaust search all the cubes containing no adjacent indexes. Then the criteria is used to test whether the cubes are conditional cubes.

4 Applications to Trivium

Trivium is a stream cipher designed in 2005 and has been selected as one of the portfolio for hardware ciphers (Profile 2) by the eSTREAM project. Though Trivium is designed to provide a flexible trade-off between speed and gate count in hardware, it also provides extremely efficient software implementation. Trivium has attracted much attention in recent public cryptographic literatures for its simplicity and very good performance, such as [26,27].

In this section, we apply our conditional searching method to Trivium to verify the validity of our method.

4.1 A brief description of Trivium

Trivium generates up to 2^{64} bits of keystream from an 80-bit secret key and an 80-bit IV. For the sake of simplicity, we give an alternative description of the algorithm different from the already existed ones. Let A, B and C be three registers of sizes 93, 84 and 111. Denoted by A^t, B^t and C^t the corresponding states at time t $(t \geq 0)$,

$$\begin{aligned} A^t &= (x_1^t, x_2^t, \cdots, x_{93}^t), \\ B^t &= (y_1^t, y_2^t, \cdots, y_{84}^t), \\ C^t &= (z_1^t, z_2^t, \cdots, z_{111}^t), \end{aligned}$$

and the three quadratic update functions are

$$\begin{aligned} x_{93}^t &= z_1^{t-1} + z_2^{t-1} \cdot z_3^{t-1} + z_{46}^{t-1} + x_{25}^{t-1}, \\ y_{84}^t &= x_1^{t-1} + x_2^{t-1} \cdot x_3^{t-1} + x_{28}^{t-1} + y_7^{t-1}, \\ z_{111}^t &= y_1^{t-1} + y_2^{t-1} \cdot y_3^{t-1} + y_{16}^{t-1} + z_{25}^{t-1}. \end{aligned}$$

The algorithm is initialized by loading an 80-bit secret key and an 80-bit IV into the 288-bit initial state, and setting all remaining bits to 0, except for z_1, z_2 and z_3,

$$\begin{aligned}(x_1^0, x_2^0, \cdots, x_{93}^0) &\leftarrow (0, \cdots, 0, k_0, \cdots, k_{79}),\\ (y_1^0, y_2^0, \cdots, y_{84}^0) &\leftarrow (0, \cdots, 0, iv_0, \cdots, iv_{79}),\\ (z_1^0, z_2^0, \cdots, z_{111}^0) &\leftarrow (1, 1, 1, 0, \cdots, 0).\end{aligned}$$

Let h be the output function. After an initialization of N rounds, in which the internal state is updated for N times, the cipher generates a keystream bit by $h(A^t, B^t, C^t)$ for each $t \geq N$.

4.2 Conditional Searching for Trivium

The 80-bit IV is loaded into the shift register B and the propagation paths are shown in Fig. 2. To find the complementary variables, we just need to control the paths ① and ② in the first few rounds. For path ①, we guarantee that the cubes are not multiplied with each other in the first iteration. According to the taps in the feedback functions, the complementary variables are iv_{i-1} and iv_{i+1} corresponding to cube variable iv_i. When the IV bits in the register C begin to be transmitted to the register A, we need to control the path ②. The complementary variables are iv_{i+14} and iv_{i+16} corresponding to cube variable iv_i. In summary, if iv_i is chosen to be a cube variable, the set of complementary variables is $\{iv_{i-1}, iv_{i+1}, iv_{i+14}, iv_{i+16}\}$.

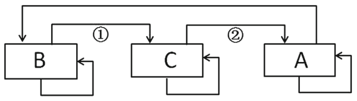

Fig. 2. Propagation paths of IV

With Algorithm 1, we know that the sizes of conditional cubes are less than 38. Then, with Algorithm 2, we can obtain all conditional cubes of size 37 and 38 in a dozen seconds on a common PC. The result, see Table 1, shows that the output of 837-round Trivium has degree strictly less than 37 over a subset of the IV bits with size 37, which agrees with [20]. The corresponding cubes

Table 1. Results of Trivium

	Maximum round/cube size	Searching space
[20]	837/37	2^{25}
Our method	837/37	$2^{12.5}$

are the same and the amount of the conditional cubes is $5945 \approx 2^{12.5}$. Before this paper, the amount of cubes need to be searched is about 2^{25} [20]. We can conclude that our conditional searching method is valid for Trivium and it has better performance.

5 Applications to Trivium-Variant Ciphers

The analysis of Trivum-like ciphers is to find a variant that will remove one of the biggest deficiencies in the Trivium design: the huge number of initial 1152 rounds, which makes the original Trivium stream cipher to have very high initial latencies when used in IoT devices. The parameters used in Trivium are subtle that a little change will affect the security a lot. Although there are some Trivium-based cryptosystems, such as Kreyvium [22], TriviA-SC [23] Bivium [24], Trivium-N [25] and so on, how do the parameters work is still a problem to be solved. In this section, we introduce two Trivium-variants named Par-Trivium and Loc-Trivium, and apply the conditional searching method to them to give some guidelines for choosing parameters according to the algebraic degree.

5.1 Applications to Par-Trivium

Par-Trivium is a variant of Trivium, where the parameters in the feedback functions of Trivium are changed. Determined by propagation paths of the IV bits, the algebraic degrees of conditional cubes are mainly associated with the taps from the register where the IV bits are loaded, and distances between the indexes of multiplied variables. In order to keep the elegant structure, the feedback functions of Par-Trivium at time t can be write as

$$\begin{aligned}
x_{93}^{t} &= z_{1}^{t-1} + z_{\alpha}^{t-1} \cdot z_{\alpha+\delta}^{t-1} + z_{46}^{t-1} + x_{25}^{t-1},\\
y_{84}^{t} &= x_{1}^{t-1} + x_{\alpha}^{t-1} \cdot x_{\alpha+\delta}^{t-1} + x_{28}^{t-1} + y_{\gamma}^{t-1},\\
z_{111}^{t} &= y_{1}^{t-1} + y_{\alpha}^{t-1} \cdot y_{\alpha+\delta}^{t-1} + y_{\beta}^{t-1} + z_{25}^{t-1}.
\end{aligned}$$

For Trivium, $\alpha = 2,\ \beta = 16,\ \gamma = 7,\ \delta = 1$. For simplicity, we denote by $R_{\alpha,\beta,\gamma,\delta}$ the maximum round that the conditional cubes of size $37 \leq n \leq 40$ have reached maximum degrees. Sometimes a part of the subscripts would be omitted and the symbol becomes $R_{\beta,\gamma}$, R_{α} and so on.

The impacts of parameters α, β, γ and δ on the algebraic degrees are summarized in the following three properties.

Property 1. *When β, γ and δ are fixed, the maximum round that the conditional cubes have reached maximum degrees decreases with the growth of α.*

It is obvious that the larger α is, the earlier the feedback values take part in the iterations and the smaller the maximum round is. For $\beta = 16,\ \gamma = 7,\ \delta = 1$, the result is listed in Table 2. We can see that the maximum rounds R_{α} is decreased with the growth of α.

Table 2. The maximum rounds R_α, when $\beta = 16,\ \gamma = 7,\ \delta = 1$

α	2	7	12	17	22	27	32	37	42	47	52	57	62	67	72	77	82
R_α	837	741	708	704	640	601	575	524	487	449	407	364	315	272	226	189	156

For Par-Trivium with parameters $(\alpha, \beta, \gamma, \delta)$, the complementary variables are $iv_{i-\delta}$, $iv_{i+\delta}$, $iv_{i+\beta-\delta-1}$ and $iv_{i+\beta+\delta-1}$ corresponding to cube variable iv_i. When $i + \beta + \delta - 1 > 79$, $iv_{i+\beta+\delta-86+\gamma}$ will be multiplied with $iv_{i+\delta-86+\gamma}$ according to the propagation paths of the IV bits. Also, when $i+\beta-\delta-1 > 79$, $iv_{i+\beta-\delta-86+\gamma}$ will be multiplied with $iv_{i-\delta-86+\gamma}$. For example, when $\alpha = 2,\ \beta = 16,\ \gamma = 7$ and $\delta = 2$, we can know that

$$
\begin{aligned}
&z_{111}^{1} = y_{16}^{0} + \cdots = iv_{11} + \cdots ,\\
&\cdots\\
&z_{111}^{4} = y_{16}^{3} + y_{2}^{3} \cdot y_{4}^{3} + \cdots = iv_{14} + iv_0 \cdot iv_2 + \cdots ,\\
&z_{111}^{5} = y_{16}^{4} + y_{1}^{4} + y_{2}^{4} \cdot y_{4}^{4} + \cdots = iv_{15} + iv_0 + iv_1 \cdot iv_3 + \cdots ,\\
&\cdots\\
&z_{111}^{88} = y_{16}^{87} + y_{1}^{87} + y_{2}^{87} \cdot y_{4}^{87} + \cdots = iv_{78} + iv_{63} + iv_{64} \cdot iv_{66} + \cdots ,\\
&z_{111}^{89} = y_{16}^{88} + y_{1}^{88} + y_{2}^{88} \cdot y_{4}^{88} + \cdots = iv_{79} + iv_{64} + iv_{65} \cdot iv_{67} + \cdots ,\\
&z_{111}^{90} = y_{16}^{89} + y_{1}^{89} + y_{2}^{89} \cdot y_{4}^{89} + \cdots = iv_{2} + iv_{65} + iv_{66} \cdot iv_{68} + \cdots ,\\
&\cdots\\
&z_{111}^{103} = y_{16}^{102} + y_{1}^{102} + \cdots = iv_{15} + iv_{78} + \cdots ,\\
&z_{111}^{104} = y_{16}^{103} + y_{1}^{103} + \cdots = iv_{16} + iv_{79} + \cdots ,\\
&z_{111}^{105} = y_{16}^{104} + y_{1}^{104} + \cdots = iv_{17} + iv_{2} + \cdots
\end{aligned}
$$

according to the feedback functions. When $t = 198$, z_{111}^{88} and z_{111}^{90} will take part in the feedback of x_{93}^{198} and iv_2 will be multiplied with iv_{63}. Also, at time $t = 213$, z_{111}^{103} and z_{111}^{105} will take part in the feedback of x_{93}^{213} and iv_2 will be multiplied with iv_{15}. Take these into consideration, we can use our conditional searching method to estimate the maximum round $R_{\alpha,\beta,\gamma,\delta}$ with respect to the conditional cubes of size $37 \le n \le 40$ and we get the following result.

Property 2. *For parameters α, β, γ and δ, let $q = 79 - \delta \mod 2\delta$ and*

$$
\beta^* = \lfloor \beta/(2\delta) \rfloor \cdot 2\delta + 1,
$$
$$
\gamma^* = \begin{cases} \delta+1+2k\delta, & 0 \le q \le \delta - 2 \\ q - \delta+2k\delta, & \delta - 1 \le q \le 2\delta - 1, \end{cases}
$$

where k is the maximum integer satisfying $\gamma \ge \gamma^*$. When α and δ are fixed, the maximum round $R_{\beta,\gamma}$ that the conditional cubes of size $37 \le n \le 40$ have reached maximum degrees, satisfies that

$$
R_{\beta,\gamma} \le R_{\beta^*,\gamma^*}
$$

and R_{β^*,γ^*} decreases with the growth of β^* or γ^*.

When α and δ are fixed, for any β and γ, we can obtain an upper bound of the number of initialization rounds such that the generated keystream bit does not achieve the maximum algebraic degree. For example, when $\alpha = 2$ and $\delta = 2$, we can get $q = 1$ and list a part of the maximum rounds for different β and γ in Table 3. We can see that in each square, the upper bound of the maximum rounds appears at the lower left. For $21 \leq \beta \leq 24$ and $9 \leq \gamma \leq 12$, it shows that $R_{\beta,\gamma} \leq R_{\beta^*,\gamma^*} = R_{21,9}$, where $\beta^* = 21$ and $\gamma^* = 9$, which is indicated by the gray part. Also, R_{β^*,γ^*} is inversely proportional to β^* or γ^*, see the numbers in bold in Table 3. Parameters β and γ for the maximum rounds in bold are the corresponding β^* and γ^*. So, in design of Trivium-Like stream cipher, it is needed to choose β and γ as big as possible, and β^* and γ^* should be avoided. While the location of the lower bound is indeterminate in spite of the fact in Table 3. It's a coincidence that the maximum round in the first column and second row is the smallest number.

Table 3. The maximum rounds $R_{\beta,\gamma}$, where $\alpha = 2$ and $\delta = 2$

β \ γ	5	6	7	8	9	10	11	12	13	17	21	25
33	**823**	810	801	812	**823**	810	801	812	**823**	**823**	**823**	**823**
32	813	811	821	821	813	811	825	821	808	797	795	795
31	809	821	814	808	811	823	814	808	813	814	816	823
30	815	827	831	815	809	826	829	817	807	807	803	803
29	**830**	821	808	820	**830**	821	809	818	**830**	**830**	**830**	**830**
28	823	809	825	829	823	811	829	829	822	818	812	804
27	791	814	824	813	809	818	824	813	810	814	816	820
26	824	833	831	815	819	833	827	817	817	817	813	809
25	**836**	824	817	826	**836**	824	817	822	**836**	**836**	**836**	**836**
24	829	823	822	833	829	817	822	833	827	825	819	815
23	812	824	822	821	805	822	822	818	796	796	816	813
22	827	829	825	825	827	829	825	825	827	827	821	817
21	**842**	830	825	832	**842**	830	825	830	**842**	**842**	**840**	**837**
20	839	831	823	829	839	827	823	828	838	834	829	821
19	-	-	-	-	-	-	-	-	-	-	-	-
18	831	825	825	827	831	827	823	827	831	831	827	823
17	**846**	842	833	838	**846**	840	833	837	**846**	**844**	**841**	**837**
16	836	840	840	830	837	841	834	828	837	837	837	833
15	-	-	-	-	-	-	-	-	-	-	-	-
14	789	829	835	833	822	826	835	833	826	825	825	825
13	**850**	848	839	842	**850**	844	839	843	**848**	**845**	**841**	**837**

- means that the conditional cubes of size $37 \leq n \leq 40$ doesn't exist.

Property 3. *Let $\alpha = 2$. When β and γ run through all possible combinations, the maximum round that the conditional cubes have reached maximum degree decreases with the growth of δ.*

It may be more persuasive to consider the relationship between the maximum round and δ if β and γ are also fixed. But when β and γ are also fixed, different

δ would lead to the sizes of the corresponding conditional cubes smaller than 37, which makes the comparison meaningless. For $\alpha = 1,\ \beta = 16,\ \gamma = 7$, the size of the conditional cubes is 38 corresponding to $\delta = 1$, but if $\delta = 6$, the size of the conditional cubes is smaller than 33. So we run through all possible β and γ to compare the maximum rounds. When $1 \leq \delta \leq 5$, the results are listed in Table 4. Here $\alpha = 2$ and three taps from register B, which are $y_1,\ y_2$ and $y_{2+\delta}$, are fixed. y_γ takes part in the feedback of register B separately and y_β takes part in the feedback of register C with $y_1,\ y_2$ and $y_{2+\delta}$ together, as

$$y_{84}^t = x_1^{t-1} + x_2^{t-1} \cdot x_{2+\delta}^{t-1} + x_{28}^{t-1} + y_\gamma^{t-1},$$
$$z_{111}^t = y_1^{t-1} + y_2^{t-1} \cdot y_{2+\delta}^{t-1} + y_\beta^{t-1} + z_{25}^{t-1}.$$

So the range of values for γ is 1 to 84 and for β is 2 to 84. When δ is given, the minimum values of β^* and γ^*, and the maximum value of rounds are determined by Property 2. But some pairs of (β, γ) aren't in consideration. For example, when $\delta = 3$, $\beta^*_{min} = 7$, $\gamma^*_{min} = 1$ and $R_{7,1} = 851$. But for $2 \leq \beta \leq 6$, we need to estimate the corresponding maximum rounds separately.

Table 4. The maximum rounds $R_{\beta,\gamma,\delta}$, where $\alpha = 2$ and k is a positive integer

δ	Taps in B	β^*	γ^*	$R_{\beta,\gamma}$
1	$\{y_1, y_2, y_3, y_\beta, y_\gamma\}$	$1+2k$	$1+2k$	$\max\{R_{\beta,\gamma} \mid 2 \leq \beta \leq 3, \gamma = 1\} = \mathrm{R}_{3,1} = 863$
2	$\{y_1, y_2, y_4, y_\beta, y_\gamma\}$	$1+4k$	$1+4k$	$\max\{R_{\beta,\gamma} \mid 2 \leq \beta \leq 5, \gamma = 1\} = \mathrm{R}_{5,1} = 861$
3	$\{y_1, y_2, y_5, y_\beta, y_\gamma\}$	$1+6k$	$1+6k$	$\max\{R_{\beta,\gamma} \mid 2 \leq \beta \leq 7, \gamma = 1\} = \mathrm{R}_{3,1} = 854$
4	$\{y_1, y_2, y_6, y_\beta, y_\gamma\}$	$1+8k$	$5+8k$	$\max\{R_{\beta,\gamma} \mid 2 \leq \beta \leq 9, 1 \leq \gamma \leq 5\} = \mathrm{R}_{9,1} = 847$
5	$\{y_1, y_2, y_7, y_\beta, y_\gamma\}$	$1+10k$	$5+10k$	$\max\{R_{\beta,\gamma} \mid 2 \leq \beta \leq 11, 1 \leq \gamma \leq 5\} = \mathrm{R}_{3,1} = 842$

The results give us some design principles for designing Trivium-Like ciphers. Ideally, the smaller the maximum rounds of NFSR-based cryptosystems such that the generated keystream bit does not achieve the maximum algebraic degree the better. So the parameters in Par-Trivium should be as big as possible on the premise of the security against other attacks. From the experiment results, we can know that the maximum round of Par-Trivium is 863 which is the worst case.

5.2 Applications to Loc-Trivium

Loc-Trivium is a variant of Trivium, where the key and the IV loading locations are changed in Trivium. In Trivium, the key bits are loaded into the register A and the IV bits are loaded into the register B. The maximum round that conditional cubes reach maximum degree is influenced mainly by propagation paths of the IV bits. So, if we change the key and IV loading locations, the maximum round will be changed. The experiments show that the maximum round is influenced mainly by the IV loading location and the corresponding taps. Here, an example is given to illustrate our findings.

Example 2. For Loc-Trivium, if we load the IV bits into the register A and the key bits into the register C, and other parameters remain unchanged, the loading procedure becomes

$$\begin{aligned}(x_1^0, x_2^0, \cdots, x_{93}^0) &\leftarrow (0, \cdots, 0, iv_0, \cdots, iv_{79}),\\ (y_1^0, y_2^0, \cdots, y_{84}^0) &\leftarrow (1, 1, 1, 0, \cdots, 0),\\ (z_1^0, z_2^0, \cdots, z_{111}^0) &\leftarrow (0, \cdots, 0, k_0, \cdots, k_{79}).\end{aligned}$$

With the tool numeric mapping, we can estimate the maximum round R of Loc-Trivium. The results are listed in the second column of Table 5. We can see that the key bits loaded into the register A and the IV bits loaded into the register B are not the best choice in view of the algebraic degree. The best choice is to load the key bits into register A and the IV bits into the register C and the maximum round is 814 smaller than 837.

Table 5. R with different key and IV loading locations

Key, IV location	R (taps unchanged)	R (taps changed)
$(A, B) \leftarrow (Key, IV)$	837	863
$(C, B) \leftarrow (Key, IV)$	828	864
$(A, C) \leftarrow (Key, IV)$	814	904
$(B, C) \leftarrow (Key, IV)$	825	910
$(B, A) \leftarrow (Key, IV)$	825	876
$(C, A) \leftarrow (Key, IV)$	825	853

If we also change the taps from the register loaded with the IV bits as shown in Sect. 5.1, we can get similar results, except the complementary variables. The feedback functions at time t turn into

$$\begin{aligned}x_{93}^t &= z_1^{t-1} + z_\alpha^{t-1} \cdot z_{\alpha+\delta}^{t-1} + z_{46}^{t-1} + x_\gamma^{t-1},\\ y_{84}^t &= x_1^{t-1} + x_\alpha^{t-1} \cdot x_{\alpha+\delta}^{t-1} + x_\beta^{t-1} + y_7^{t-1},\\ z_{111}^t &= y_1^{t-1} + y_\alpha^{t-1} \cdot y_{\alpha+\delta}^{t-1} + y_{16}^{t-1} + z_{25}^{t-1}.\end{aligned}$$

The complementary variables are $iv_{i-\delta}$, $iv_{i+\delta}$, $iv_{i+\beta-\delta-1}$ and $iv_{i+\beta+\delta-1}$ corresponding to the cube variable iv_i. When $i+\beta+\delta-1 > 79$, $iv_{i+\beta+\delta-113+\gamma}$ will be multiplied with $iv_{i+\delta-113+\gamma}$ according to the propagation paths of the IV bits. Also, when $i+\beta-\delta-1 > 79$, $iv_{i+\beta-\delta-113+\gamma}$ will be multiplied with $iv_{i-\delta-113+\gamma}$.

In the third column of Table 5, we list the maximum rounds R when the corresponding parameters are also changed. The result shows that Loc-Trivium with the key loaded in the register B and IV loaded in the register C has the maximum rounds 910, which is the worst case. The corresponding feedback functions are

$$x_{93}^t = z_1^{t-1} + z_2^{t-1} \cdot z_3^{t-1} + \mathbf{z_3^{t-1}} + x_{25}^{t-1},$$
$$y_{84}^t = x_1^{t-1} + x_2^{t-1} \cdot x_3^{t-1} + x_{28}^{t-1} + y_7^{t-1},$$
$$z_{111}^t = y_1^{t-1} + y_2^{t-1} \cdot y_3^{t-1} + y_{16}^{t-1} + \mathbf{z_1^{t-1}}.$$

Here some tap positions are the same. Ensure that all tap positions are distinct, we obtain that the maximum rounds is 906 and the corresponding feedback functions are

$$x_{93}^t = z_1^{t-1} + z_2^{t-1} \cdot z_3^{t-1} + \mathbf{z_5^{t-1}} + x_{25}^{t-1},$$
$$y_{84}^t = x_1^{t-1} + x_2^{t-1} \cdot x_3^{t-1} + x_{28}^{t-1} + y_7^{t-1},$$
$$z_{111}^t = y_1^{t-1} + y_2^{t-1} \cdot y_3^{t-1} + y_{16}^{t-1} + \mathbf{z_4^{t-1}}.$$

The results show that the key and IV loading locations play important roles in the algebraic degree of NFSR-based cryptosystems. When a new cipher with the similar structure is designed, both the parameters and the key and IV loading locations should be taken into account. The worst case that the maximum rounds is 910 or 906, which should be avoided to be resistant to cube attacks or cube tests when new ciphers are designed.

6 Conclusions

In this paper, we have shown a new cube searching method. The main idea is to reduce the searching space through controlling the propagation of the IV bits. With our cube searching method and the algebraic degree estimated method numeric mapping, we can confirm the maximum numbers of initialization rounds of some NFSR-based cryptosystems such that the generated keystream bit does not achieve the maximum algebraic degree. As illustrations, we applied our method to Trivium to verify the validity, our searching space is much smaller than that of the existing results. We also applied our method to two Trivium-variants, named Par-Trivium and Loc-Trivium, and we can get an upper bound of the maximum initialization rounds.

We believe that our method is useful in both cryptanalysis and design of NFSR-based cryptosystems. In design of Trivium-Like stream cipher, it is needed to choose α, β, γ and δ as big as possible on the premise of the security against other attacks, and β^* and γ^* should be avoided. The experiments show that the maximum round of Par-Trivium is 863 which is the worst case. For the key and IV loading locations, if other parameters are not changed, the loading locations used in Trivium are the worst case, but it doesn't threaten the security. If other parameters are also changed, the worst and best choices are also given. For Loc-Trivium, the maximum initialization round of all considered Trivium-variants is 910 which is the worst case and should be avoided to be resistant to cube attacks or cube tests.

Acknowledgment. The authors would like to thank the anonymous reviewers for their comments and suggestions which significantly improve the quality and presentation of this paper. This work was supported by the National Natural Science Foundation of China (Grant No. 61672516), the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDA06010701), and the Fundamental Theory and Cutting Edge Technology Research Program of Institute of Information Engineering, CAS (Grant No. Y7Z0331102).

References

1. Dinur, I., Güneysu, T., Paar, C., Shamir, A., Zimmermann, R.: An experimentally verified attack on full Grain-128 using dedicated reconfigurable hardware. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 327–343. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_18
2. Dinur, I., Morawiecki, P., Pieprzyk, J., Srebrny, M., Straus, M.: Cube attacks and cube-attack-like cryptanalysis on the round-reduced Keccak sponge function. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 733–761. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46800-5_28
3. Dinur, I., Shamir, A.: Cube attacks on tweakable black box polynomials. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 278–299. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_16
4. Dinur, I., Shamir, A.: Breaking Grain-128 with dynamic cube attacks. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 167–187. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21702-9_10
5. Knudsen, L.R.: Truncated and higher order differentials. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 196–211. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-60590-8_16
6. Moriai, S., Shimoyama, T., Kaneko, T.: Higher order differential attack of a CAST cipher. In: Vaudenay, S. (ed.) FSE 1998. LNCS, vol. 1372, pp. 17–31. Springer, Heidelberg (1998). https://doi.org/10.1007/3-540-69710-1_2
7. Huang, S., Wang, X., Xu, G., Wang, M., Zhao, J.: Conditional cube attack on reduced-round Keccak sponge function. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10211, pp. 259–288. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56614-6_9
8. Li, Z., Dong, X., Wang, X.: Conditional cube attack on round-reduced ascon. IACR Trans. Symmetric Cryptol. **1**, 175–202 (2017)
9. Bi, W., Li, Z., Dong, X., Li, L., Wang, X.: Conditional cube attack on round-reduced River Keyak. Des. Codes Crypt. **86**(6), 1295–1310 (2018)
10. Courtois, N.T.: Fast algebraic attacks on stream ciphers with linear feedback. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 176–194. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_11
11. Courtois, N.T., Pieprzyk, J.: Cryptanalysis of block ciphers with overdefined systems of equations. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 267–287. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-36178-2_17
12. Knudsen, L., Wagner, D.: Integral cryptanalysis. In: Daemen, J., Rijmen, V. (eds.) FSE 2002. LNCS, vol. 2365, pp. 112–127. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45661-9_9
13. Boura, C., Canteaut, A., De Cannière, C.: Higher-order differential properties of Keccak and *Luffa*. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 252–269. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21702-9_15

14. Canteaut, A., Videau, M.: Degree of composition of highly nonlinear functions and applications to higher order differential cryptanalysis. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 518–533. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-46035-7_34
15. Todo, Y.: Structural evaluation by generalized integral property. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 287–314. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46800-5_12
16. Todo, Y., Isobe, T., Hao, Y., Meier, W.: Cube attacks on non-blackbox polynomials based on division property. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10403, pp. 250–279. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63697-9_9
17. De Cannière, C., Preneel, B.: Trivium. In: Robshaw, M., Billet, O. (eds.) New Stream Cipher Designs. LNCS, vol. 4986, pp. 244–266. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-68351-3_18
18. Ågren, M., Hell, M., Johansson, T., Meier, W.: Grain-128a: a new version of Grain-128 with optional authentication. Int. J. Wirel. Mob. Comput. **5**(1), 48–59 (2011)
19. Wu, H.: ACORN: A Lightweight Authenticated Cipher (v3) (2016). http://competitions.cr.yp.to/round3/acornv3.pdf
20. Liu, M.: Degree evaluation of NFSR-based cryptosystems. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10403, pp. 227–249. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63697-9_8
21. Hell, M., Johansson, T., Maximov, A., Meier, W.: The Grain family of stream ciphers. In: Robshaw, M., Billet, O. (eds.) New Stream Cipher Designs. LNCS, vol. 4986, pp. 179–190. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-68351-3_14
22. Canteaut, A., et al.: Stream ciphers: a practical solution for efficient homomorphic-ciphertext compression. In: Peyrin, T. (ed.) FSE 2016. LNCS, vol. 9783, pp. 313–333. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-52993-5_16
23. Chakraborti, A., Chattopadhyay, A., Hassan, M., Nandi, M.: TriviA: a fast and secure authenticated encryption scheme. In: Güneysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 330–353. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48324-4_17
24. Raddum, H.: Cryptanalytic results on Trivium. eSTREAM, ECRYPT Stream Cipher Project, Report, 39 (2006)
25. Schilling, T.E., Raddum, H.: Analysis of Trivium using compressed right hand side equations. In: Kim, H. (ed.) ICISC 2011. LNCS, vol. 7259, pp. 18–32. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31912-9_2
26. Fu, X., Wang, X., Dong, X., Meier, W.: A Key-recovery Attack on 855-round Trivium. IACR Cryptology ePrint Archive 2018, 198 (2018)
27. Fouque, P.-A., Vannet, T.: Improving key recovery to 784 and 799 rounds of Trivium using optimized cube attacks. In: Moriai, S. (ed.) FSE 2013. LNCS, vol. 8424, pp. 502–517. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43933-3_26