# Degree Evaluation of NFSR-Based Cryptosystems

Meicheng Liu(✉)

State Key Laboratory of Information Security, Institute of Information Engineering,
Chinese Academy of Sciences, Beijing 100093, People's Republic of China
meicheng.liu@gmail.com

**Abstract.** In this paper, we study the security of NFSR-based cryptosystems from the algebraic degree point of view. We first present a general framework of iterative estimation of algebraic degree for NFSR-based cryptosystems, by exploiting a new technique, called *numeric mapping*. Then based on this general framework we propose a concrete and efficient algorithm to find an upper bound on the algebraic degree for Trivium-like ciphers. Our algorithm has linear time complexity and needs a negligible amount of memory. As illustrations, we apply it to Trivium, Kreyvium and TriviA-SC, and reveal various upper bounds on the algebraic degree of these ciphers by setting different input variables. By this algorithm, we can make use of a cube with any size in cube testers, which is generally believed to be infeasible for an NFSR-based cryptosystem before. Due to the high efficiency of our algorithm, we can exhaust a large set of the cubes with large size. As such, we obtain the best known distinguishing attacks on reduced Trivium and TriviA-SC as well as the first cryptanalysis of Kreyvium. Our experiments on Trivium show that our algorithm is not only efficient in computation but also accurate in estimation of attacked rounds. The best cubes we have found for Kreyvium and TriviA-SC are both of size larger than 60. To the best of our knowledge, our tool is the first formalized and systematic one for finding an upper bound on the algebraic degree of an NFSR-based cryptosystem, and this is the first time that a cube of size beyond practical computations can be used in cryptanalysis of an NFSR-based cryptosystem. It is also potentially useful in the future applications to key recovery attacks and more cryptographic primitives.

**Keywords:** Nonlinear feedback shift register · Stream cipher · Distinguishing attack · Cube tester · Trivium · Kreyvium · TriviA-SC

## 1 Introduction

A nonlinear feedback shift register (NFSR) is a common component in modern cryptographic primitives, especially in radio-frequency identification devices

This work was supported by the National Natural Science Foundation of China (Grant Nos. 61672516, 61303258 and 61379139) and the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDA06010701.

J. Katz and H. Shacham (Eds.): CRYPTO 2017, Part III, LNCS 10403, pp. 227–249, 2017.
DOI: 10.1007/978-3-319-63697-9_8

(RFID) and wireless sensor networks applications. NFSRs are known to be more resistant to cryptanalytic attacks than linear feedback shift registers (LFSRs). Built on NFSRs are many well known lightweight cryptographic algorithms, including the stream ciphers Trivium [8,10] and Grain [1,27,28] that have been selected in the final eSTREAM portfolio of hardware-oriented stream ciphers, the authenticated cipher ACORN [44] that has been selected as one of the third-round candidates in the CAESAR competition, the block cipher family KATAN/KTANTAN [9], and the hash function Quark [4,5]. Among them, Trivium has attracted the most attention for its simplicity and performance, while it shows remarkable resistance to cryptanalysis. Inspired by the design of Trivium, a number of various cryptographic algorithms have been successively developed, for instance the block cipher family KATAN/KTANTAN, the authenticated cipher ACORN and the stream ciphers Kreyvium [11] and TriviA-SC [13].

Most cryptographic primitives, including NFSR-based cryptosystems, can be described by tweakable Boolean functions, which contain both secret variables (*e.g.*, key bits) and public variables (*e.g.*, plaintext bits or IV bits). The algebraic degree of these Boolean functions plays an important role in the security of the corresponding primitives. In fact, a cryptographic primitive with low algebraic degree is vulnerable to many known attacks, such as higher order differential attacks [30,32,35], algebraic attacks [15–18], cube attacks [19–22], and integral attacks [31].

For NFSR-based cryptosystems, cube attacks and higher order differential attacks are the most powerful cryptanalytic tools among the known attacks. The best known key recovery attacks faster than an exhaustive search on Trivium are cube attacks on its variant when the initialization is reduced to 799 rounds out of 1152 [21,26], and the best known distinguishing attacks on Trivium are reduced to 839 rounds derived by cube testers [3,33]. Note that here are not included the possible key recovery attacks with unknown probability, such as [41], or the attacks for a small percentage of weak keys, such as [29]. The weaknesses in the cipher Grain-128 against cube testers [2,39] partially leads to the design of Grain-128a [1]. Actually, the full Grain-128 was broken in theory by dynamic cube attacks [19,22]. All of these attacks exploit low-degree relations of the tweakable Boolean functions formed by the cryptosystems, that is, low-degree relations between the IV bits and keystream bits.

It is difficult to compute the exact value of the algebraic degree for modern cryptographic primitives. After the development of cryptanalysis in the past three decades, several theoretical tools have been developed to estimate the upper bound on the algebraic degree of iterated permutations, and concurrently exploited to attack iterated ciphers [6,7,12,40].

Yet for NFSR, there are few tools for estimating its algebraic degree, besides symbolic computation and statistical analysis. The known techniques highly depends on computational capabilities, and the cryptanalytic results are limited by existing computational resources. For instance, thus far the cubes with size larger than 54 have never been utilized in cryptanalysis of an NFSR-based cryptosystem, in either cube attacks or cube testers. To gain better attacks, the cryptanalysts have to utilize extremely the computational resources, *e.g.*, using dedicated reconfigurable hardware [19]. This usually requires high financial cost or high energy consumption. While dynamic cube attacks [19,22] can reach much higher attack complexity, they are still limited by the size of the cubes.

### 1.1 Our Contributions

In this paper, we devote our attention to evaluating the algebraic degree of NFSR-based cryptosystems. For the conquest of the existing limitation as mentioned above, we exploit a new technique, called *numeric mapping*, to iteratively estimate the upper bound on the algebraic degree of the internal states of an NFSR. Based on this new tool, we develop an algorithm for estimating the algebraic degree of NFSR-based cryptosystems.

As an illustration, we refine and apply our algorithm to Trivium-like ciphers, including Trivium, Kreyvium and TriviA-SC. Trivium uses an 80-bit key and an 80-bit IV, while Kreyvium and TriviA-SC both use a 128-bit key and a 128-bit IV. These three ciphers all have 1152 rounds of initialization. Our refined algorithm gives an upper bound on the algebraic degree of a Trivium-like cipher over a given set of input variables with any size, *e.g.*, all the key and IV bits, all or part of the IV bits. It has linear time complexity in the number of initialization rounds, and needs a negligible amount of memory. In other words, it is almost as fast as the cipher (up to at most a factor of some constant). Further, by this algorithm we perform several experiments on round-reduced Trivium, Kreyvium and TriviA-SC, and obtain various upper bounds on the algebraic degree by setting different input variables. As a result, we confirm that the maximum numbers of initialization rounds of Trivium, Kreyvium and TriviA-SC such that the generated keystream bit does not achieve maximum algebraic degree are at least 907, 982 and 1121 (out of the full 1152 rounds) respectively when taking all the key and IV bits as input variables; these numbers of rounds turn out to be 793, 862 and 987 while taking all the IV bits as input variables.

We further apply our algorithm to take advantage of the cubes with large size in cube testers, which is considered to be impossible for an NFSR-based cryptosystem in the literatures. In the experiments, we set the key bits as symbolic constants, *i.e.*, the algebraic degree of any key bit is considered to be 0 on the cube variables. This is consistent with a distinguisher in the setting of unknown key. Since our algorithm is very fast, we can exhaust all the cubes of size $37 \leq n \leq 40$ that contain no adjacent indexes for Trivium in a dozen minutes on a common PC. The total amount of such cubes is about $2^{25}$. Before this paper, it needs around $c2^{62}$ cipher operations to test all those cubes, and the confidence of the test depends on $c$; while our algorithm is deterministic. We then find a cube of size 37 over which the algebraic degree of the keystream bit of 837-round Trivium is strictly less than 37. We also verify this result by performing experiments on 100 random keys. The minimum number of rounds that the sum over this cube, called superpoly in cube attacks and cube testers, is not zero-constant is detected to be 839 in our experiments, which implies that our algorithm is not only efficient in computation but also accurate in estimation of attacked rounds. Our experiments show that this cube can also be used to distinguish 842-round Trivium. All the cubes of size $61 \leq n \leq 64$ that contain no adjacent indexes for Kreyvium and TriviA-SC are exhausted in a few hours. The total amount of such cubes is about $2^{30}$. By the conventional methods, it needs around $c2^{91}$ cipher operations. The best cube we have found for Kreyvium is of size 61, which can be used to distinguish 872-round Kreyvium. The best cubes we have found for TriviA-SC and its successor are respectively of size 63 and size 61, for distinguishing 1035 rounds and 1047 rounds respectively. To the best of our knowledge, this is the first time[1] that a cube of size larger than 60 can be used in the attack on an NFSR-based cryptosystem.

As such, we obtain the best distinguishing attacks for the stream ciphers Trivium and TriviA-SC so far and the first outside cryptanalysis of Kreyvium. Our results are summarized in Table 1 with the comparisons of the previous attacks. Note here that this table does not include the distinguishers worse than an exhaustive search or for a small percentage of weak keys. We detail the discussions of related work in the following.

**Table 1.** Distinguishing attacks on Trivium, Kreyvium and TriviA-SC

| Cipher | #Rounds | Complexity | Ref. |
|---|---|---|---|
| Trivium | 790 | $2^{30}$ | [3] |
| | 798 | $2^{25}$ | [29] |
| | 806 | $2^{44}$ | [39] |
| | 829 | $2^{53}$ | [38] |
| | 830 | $2^{39}$ | [43] |
| | 839 | $2^{37}$ | [33] |
| | 842 | $2^{39}$ | Sect. 4 |
| Kreyvium | 872 | $2^{61}$ | Sect. 4 |
| TriviA-SC (v1) | 930 | $2^{36}$ | [38] |
| | 1035 | $2^{63}$ | Sect. 4 |
| TriviA-SC (v2) | 950 | $2^{36}$ | [38] |
| | 1047 | $2^{61}$ | Sect. 5 |
| Simplified TriviA-SC | 1152 | $2^{120}$ | [45] |
| | 1152 | $2^{63}$ | Sect. 4 |

[1] In parallel and independently with our work, large cubes have also been exploited by Todo *et al.* [41] in the attacks on NFSR-based cryptosystems, such as Trivium, Grain-128a and ACORN.

### 1.2 Related Work

**Upper Bound on Algebraic Degree.** At EUROCRYPT 2002, Canteaut and Videau [12] developed a theory to find an upper bound on the algebraic degree of a composite function using the Walsh spectrum, and applied it to higher order differential cryptanalysis on Feistel block ciphers and especially on a generalization of MISTY1. This theory was further improved by Boura *et al.* [6,7] in recent years with applications to cryptanalysis of several block ciphers and hash functions, including Rijndael-256 and Keccak. These theories of estimating algebraic degree are suitable for iterated ciphers. Similarly, our work is started by an upper bound on the algebraic degree of a composite function, but without using the Walsh spectrum and based on a simple fact.

More recently, at EUROCRYPT 2015, Todo [40] discovered a new tool for searching upper bound on the algebraic degree of SPN and Feistel ciphers by introducing the division property with applications to integral cryptanalysis of various iterated cryptographic primitives. The bit-based division property proposed by Todo and Morii in [42] is more relevant to our work. In parallel with our work, this tool has been exploited by Todo *et al.* [41] for estimating the algebraic degree of NFSR-based cryptosystems, including Trivium, Grain-128a and ACORN, and applied to cube attacks on these ciphers. Nevertheless, our idea is still essentially different with that of division property. In some ways, the tool based on division property is limited by the number of rounds and the size of input variables, due to its high time complexity. The bound found by division property is possibly more precise, while our tool is much faster and has no such limitations.

**Attacks on Trivium-Like Ciphers.** It is worth noticing that all but the attacks of [45] listed in Table 1 are cube tester, which is a variant of higher order differential attacks and was first introduced by Aumasson *et al.* in [3]. Cube testers are useful not only in distinguishing attacks but also in key recovery attacks, *e.g.*, dynamic cube attacks [19,22] and cube-attack-like cryptanalysis [20].

Before the work of Aumasson *et al.*, Trivium (designed by Cannière and Preneel [8,10] in 2006) had already attracted a lot of similar cryptanalysis, especially for chosen IV statistical attacks, *e.g.*, [23,24,37]. After the effort of cryptanalysts in the past ten years, the cryptanalysis of Trivium seems to be approaching a bottleneck, if not the summit. Several cube distinguishers under different statistical models reach around 830 rounds, *e.g.*, [33,38,43]. Though our distinguisher for Trivium does not improve the previous ones much, our technique for finding cubes is novel and gives a new and global view on cube cryptanalysis of Trivium.

In addition, Knellwolf *et al.* [29] showed distinguishers on 868-round and 961-round Trivium respectively for $2^{31}$ and $2^{26}$ weak keys both with complexity of $2^{25}$. The key recovery attacks are also well studied for Trivium. In [21], Dinur and Shamir described a practical full key recovery on Trivium reduced to 767 rounds, using cube attacks. Afterwards, Fouque and Vannet [26] improved the cube attacks on Trivium, and provided a practical full key recovery after 784 rounds and a full key recovery after 799 rounds with complexity of $2^{62}$. Recently, Todo *et al.* [41] proposed a possible key recovery after 832 rounds, in which one bit information of the key can be retrieved with unknown probability in around $2^{77}$. Besides, Maximov and Biryukov [34] presented a state recovery attack on the full cipher with time complexity around $c2^{83.5}$, where $c$ is the complexity of solving a system of linear equations with 192 variables.

TriviA-SC [13] is a stream cipher designed by Chakraborti *et al.* at CHES 2015 for using in the authenticated encryption scheme TriviA, which was selected as a second-round candidate in the CAESAR competition but was not retained for the third round. Its successor, TriviA-SC (v2) [14], retains the same design and only differs in flipping all but three bits of the constants loaded to the initial internal state. Sarkar *et al.* [38] showed cube distinguishers with complexity of $2^{36}$ on both versions of TriviA-SC reduced to 930 rounds and 950 rounds respectively. We improve these distinguishers to 1035 rounds and 1047 rounds respectively. The work of [45] by Xu *et al.* shows a linear distinguisher with complexity of $2^{120}$ for the full 1152 rounds of a simplified variant of TriviA-SC in which the unique nonlinear term of the output function is removed. As shown in Table 1, we cut down their complexity from $2^{120}$ to $2^{63}$ for this simplified TriviA-SC.

Kreyvium is a variant of Trivium with 128-bit security, designed by Canteaut *et al.* at FSE 2016 for efficient homomorphic-ciphertext compression [11]. As far as we know, this paper proposes the first cryptanalysis of Kreyvium.

### 1.3 Organization

The rest of this paper is structured as follows. In Sect. 2, the basic definitions and notations are provided. Section 3 shows the general framework of our algorithm for estimating algebraic degree of NFSR-based cryptosystems. We propose in Sect. 4 a concrete algorithm for finding an upper bound on the algebraic degree of Trivium-like ciphers with applications to Trivium, Kreyvium and TriviA-SC, while Sect. 5 further presents an improved algorithm with applications to TriviA-SC. Section 6 concludes the paper.

## 2 Preliminaries

**Boolean Functions and Algebraic Degree.** Let $\mathbb{F}_2$ denote 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 $\mathbb{F}_2$. Denote by $\mathbb{B}_n$ the set of all $n$-variable Boolean functions. An $n$-variable Boolean function $f$ can be uniquely represented as a multivariate polynomial over $\mathbb{F}_2$,

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

called the algebraic normal form (ANF). The algebraic degree of $f$, denoted by $\deg(f)$, is defined as $\max\{wt(c) \mid a_c \neq 0\}$, where $wt(c)$ is the Hamming weight of $c$. Let $g_i$ $(1 \leq i \leq m)$ be Boolean functions on $n$ variables. We denote $\deg(G) = (\deg(g_1), \deg(g_2), \cdots, \deg(g_m))$, for $G = (g_1, g_2, \cdots, g_m)$.

**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 the sum of terms which are supersets of $I$ and terms that miss at least one variable from $I$,

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

where $f_S(I)$ is called the superpoly of $I$ in $f$. The basic idea of cube testers is that the symbolic sum of all the derived polynomials obtained from the function $f$ by assigning all the possible values to the subset of variables in the term $t_I$ is exactly $f_S(I)$. Cube testers work by evaluating superpolys of carefully selected terms $t_I$ which are products of public variables (*e.g.*, IV bits), and trying to distinguish them from a random function. Especially, the superpoly $f_S(I)$ is equal to a zero constant, if the algebraic degree of $f$ in the variables from $I$ is less than the size of $I$. In this paper, we mainly focus on this case. For more details of cube testers, we refer to [3].

**Nonlinear Feedback Shift Registers.** Nonlinear feedback shift registers (NFSRs) are the basic components of cryptographic primitives, especially of stream ciphers. Each time the system is clocked, the internal state is shifted right, and the new left bit is computed from the previous state by a nonlinear function $f$. The feedback bit is computed as

$$s_{t+1} = f(s_t, \cdots, s_{t-n+1}),$$

where $f$ can be any function in $n$ variables. According to implementation purposes, the most useful case is the binary case, in which each cell contains a bit, and $f$ is a Boolean function. In this paper, we focus on this binary case. For more details of NFSRs, we refer to [25].

## 3 An Iterative Method for Estimating Algebraic Degree of NFSR-Based Cryptosystems

Compared with other types of cryptographic primitives, such as Feistel and SPN ciphers, an NFSR-Based Cryptosystem usually updates less bits each round and needs more rounds to ensure its security, and its algebraic degree is more irregular. Maybe due to this reason, besides experimental analysis there are few theoretical tools to estimate algebraic degree of NFSR-Based cryptosystems.

We will show in this section a general idea for iteratively estimating algebraic degree of NFSR-based cryptosystems. We first present a basic fact on the degree of a composite function, and then exploit it to estimate degrees of the internal states and outputs of NFSR-based cryptosystems.

Let $f(x_1, x_2, \cdots, x_m) = \bigoplus_{c=(c_1,\cdots,c_m)\in\mathbb{F}_2^m} a_c \prod_{i=1}^m x_i^{c_i}$ be a Boolean function on $m$ variables. We define the following mapping, called *numeric mapping* and denoted by $\texttt{DEG}$,

$$\begin{aligned} \texttt{DEG}:\ & \mathbb{B}_m \times \mathbb{Z}^m \rightarrow \mathbb{Z}, \\ & (f, D) \mapsto \max_{a_c \neq 0}\{\sum_{i=1}^m c_i d_i\}, \end{aligned}$$

where $D = (d_1, d_2, \cdots, d_m)$ and $a_c$'s are coefficients of algebraic normal form of $f$ as defined previously.

Let $g_1, g_2, \cdots, g_m$ be Boolean functions on $n$ variables, $G = (g_1, g_2, \cdots, g_m)$ and $\deg(G) = (\deg(g_1), \deg(g_2), \cdots, \deg(g_m))$. The numeric degree of the composite function $h = f \circ G$ is defined as $\texttt{DEG}(f, \deg(G))$, denoted by $\texttt{DEG}(h)$ for short. We call $\texttt{DEG}(f, D)$ a super numeric degree of $h$ if $d_i \geq \deg(g_i)$ for all $1 \leq i \leq m$, where $D = (d_1, d_2, \cdots, d_m)$. We can check that the algebraic degree of $h$ is always less than or equal to the numeric degree of $h$, *i.e.*,

$$\deg(h) = \deg(f(g_1, g_2, \cdots, g_m)) \leq \texttt{DEG}(h) = \max_{a_c \neq 0}\{\sum_{i=1}^m c_i \deg(g_i)\}.$$

**Proposition 1.** *The algebraic degree of a composite function is less than or equal to its numeric degree.*

An NFSR-based cryptosystem usually consists of an update function $g$ and an output function $f$. The internal state is updated by the update function $g$, while the output bit is generated by the output function $f$ after an initialization of a sufficient number of rounds. To make the implementation efficient, the update function and output function usually have extremely sparse terms, *e.g.*, Trivium [8,10] and Grain [1,27,28]. Even though these functions are simple, there are few tools to exactly compute their algebraic degrees after updating the internal state by a sufficient number of rounds. A straightforward way to achieve this is to calculate the algebraic normal form, but it easily becomes out of memory as the number of rounds increases. A more efficient method is to test the coefficients of the algebraic normal form by statistical analysis, but it highly depends on the computational power and is limited by computational time. To overcome these limitations of computational resources, we exploit the numeric mapping to estimate the algebraic degree.

**Corollary 2.** *Denote by $s^{(t)}$ the internal state of an NFSR-based cryptosystem at $t$-th round, and let $g$ and $f$ be the update function and output function respectively. Then the algebraic degrees of the updated bit and output bit are respectively less than or equal to their numeric degrees, i.e.,* $\texttt{DEG}(g, \deg(s^{(t)}))$ *and* $\texttt{DEG}(f, \deg(s^{(t)}))$.

*Example 1.* Let $x_t = x_{t-2}x_{t-7} + x_{t-4}x_{t-5} + x_{t-8}$ be the update function of an NFSR with size 8. For $t = 16$, we have

$$x_{16} = x_{14}x_9 + x_{12}x_{11} + x_8.$$

We can iteratively compute

$$\begin{aligned}
x_9 &= x_2x_7 + x_4x_5 + x_1,\\
x_{11} &= x_2x_4x_7 + x_1x_4 + x_4x_5 + x_6x_7 + x_3,\\
x_{12} &= x_3x_5x_8 + x_2x_5 + x_5x_6 + x_7x_8 + x_4,\\
x_{14} &= x_2x_3x_7x_8 + x_2x_5x_6x_7 + x_3x_4x_5x_8 + x_3x_5x_7x_8\\
&\quad + x_1x_3x_8 + x_1x_5x_6 + x_2x_4x_5 + x_2x_5x_7 + x_4x_5x_6\\
&\quad + x_5x_6x_7 + x_1x_2 + x_2x_7 + x_4x_7 + x_7x_8 + x_6.
\end{aligned}$$

Then by numeric mapping, we have

$$\begin{aligned}
\texttt{DEG}(x_{16}) &= \max\{\deg(x_{14}) + \deg(x_9), \deg(x_{12}) + \deg(x_{11}), \deg(x_8)\}\\
&= \max\{4+2, 3+3, 1\}\\
&= 6.
\end{aligned}$$

We can verify that $\deg(x_{16}) = 6$ by calculating the algebraic normal form of $x_{16}$. As a matter of fact, we can also check that $\texttt{DEG}(x_t) = \deg(x_t)$ for all $t < 16$. This fact implies that we can get an accurate estimation of the algebraic degree of $x_{16}$ by iteratively using numeric mapping starting at the beginning, without computations of the algebraic normal forms of internal bits.

The case that the numeric degree equals the algebraic degree usually happens when the intermediate variables appearing in the same nonlinear terms are independent. This scenario is reasonable for an ideal cryptosystem. For a concrete cipher, the numeric degree might be equal or close to the algebraic degree if we eliminate or reduce the dependent relationship between the intermediate variables.

**Algorithm 1.** Estimation of Degree of NFSR-Based Cryptosystems

**Require:** Given the ANFs of the internal state $s^{(0)}$, the ANFs of the update function $G$ and output function $f$, and the set of variables $X$.

```
Set $D^{(0)}$ and $E^{(0)}$ to $\deg(s^{(0)}, X)$;
For $t$ from 1 to $N$ do:
    Compute $D^{(t)} = \texttt{DegEst}(G, E^{(t-1)})$;
    Set $E^{(t)}$ to $(D^{(0)}, D^{(1)}, \cdots, D^{(t)})$;
Return $\texttt{DegEst}(f, E^{(N)})$.
```

The algebraic degrees of output bits and the internal states can be estimated iteratively for NFSR-based cryptosystems. We describe this estimation in Algorithm 1. In the algorithm, $s^{(0)} = (s_1^{(0)}, s_2^{(0)}, \cdots, s_n^{(0)})$

denotes the internal state at time 0 with size $n$, and $\deg(s^{(0)}, X) = (\deg(s_1^{(0)}, X), \deg(s_2^{(0)}, X), \cdots, \deg(s_n^{(0)}, X))$, where the notation $\deg(s_i^{(0)}, X)$ denotes the algebraic degree of $s_i^{(0)}$ with $X$ as variables. Especially, $\deg(0, X) = -\infty$, and $\deg(c, X) = 0$ for any nonzero $c$ containing no variable in $X$. The update function $G$ is written as vectorial Boolean functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$, where a few bits of input are updated and the rest of the bits are shifted. `DegEst` is a procedure for estimating algebraic degree. The output of this algorithm gives an upper bound on algebraic degree of the output of a given NFSR-based cryptosystem when setting $\texttt{DegEst}(\cdot, E^{(t)})$ to $\texttt{DEG}(\cdot, D^{(t)})$. This is based on the fact that $\deg(g(s^{(t)})) \leq \texttt{DEG}(g, \deg(s^{(t)})) \leq \texttt{DEG}(g, \texttt{DEG}(s^{(t)}))$ according to Corollary 2.

Now we have given a general framework of iterative estimation of algebraic degree of NFSR-Based Cryptosystems. To reach a sharper upper bound, we use a more delicate `DegEst` rather than `DEG` in Algorithm 1. We will show later the applications to Trivium-like ciphers, and the experimental results show that our estimated degree is very close to the real value of algebraic degree.

## 4 Applications to Trivium-Like Ciphers

In this section, we first briefly describe a generic view of a Trivium-like cipher to capture various cryptographic algorithms such as Trivium, TriviA-SC and Kreyvium. Then, based on our observations on the update functions of this kind of ciphers, we formalize and develop a linear-time algorithm for finding an upper bound on the algebraic degree of a Trivium-like cipher. Finally, we apply our algorithm to analyze the security of the ciphers Trivium, TriviA-SC and Kreyvium.

### 4.1 A Brief Description of Trivium-Like Ciphers

Let $A$, $B$ and $C$ be three registers with sizes of $n_A, n_B$ and $n_C$, denoted by $A_t$, $B_t$ and $C_t$ their corresponding states at clock $t$,

$$A_t = (x_t, x_{t-1}, \cdots, x_{t-n_A+1}), \tag{1}$$

$$B_t = (y_t, y_{t-1}, \cdots, y_{t-n_B+1}), \tag{2}$$

$$C_t = (z_t, z_{t-1}, \cdots, z_{t-n_C+1}), \tag{3}$$

and respectively updated by the following three quadratic functions,

$$x_t = z_{t-r_C} \cdot z_{t-r_C+1} + \ell_A(s^{(t-1)}), \tag{4}$$

$$y_t = x_{t-r_A} \cdot x_{t-r_A+1} + \ell_B(s^{(t-1)}), \tag{5}$$

$$z_t = y_{t-r_B} \cdot y_{t-r_B+1} + \ell_C(s^{(t-1)}), \tag{6}$$

where $1 \leq r_\lambda < n_\lambda$ for $\lambda \in \{A, B, C\}$ and $\ell_A, \ell_B$ and $\ell_C$ are linear functions. We denote $A_t[i] = x_i$, $B_t[i] = y_i$ and $C_t[i] = z_i$, and define $g_A^{(t)} = z_{t-r_C} \cdot z_{t-r_C+1}$, $g_B^{(t)} = x_{t-r_A} \cdot x_{t-r_A+1}$ and $g_C^{(t)} = y_{t-r_B} \cdot y_{t-r_B+1}$. The internal state, denoted by

$s^{(t)}$ at clock $t$, consists of the three registers $A, B, C$, that is, $s^{(t)} = (A_t,\ B_t,\ C_t)$. Let $f$ 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 $f(s^{(t)})$ for each $t \geq N$.

Trivium and TriviA-SC exactly fall into this kind of ciphers. As mentioned earlier, TriviA-SC and its successor TriviA-SC (v2) only differ in the constants loaded to the initial internal state. Hereinafter, TriviA-SC means its both versions, if not specified. Kreyvium is a variant of Trivium with 128-bit security. Compared with Trivium, Kreyvium uses two extra registers $(K^*, V^*)$ without updating but shifting, *i.e.*, $s^{(t)} = (A_t,\ B_t,\ C_t, K^*, V^*)$, and add a single bit of $(K^*, V^*)$ to each of $\ell_A$ and $\ell_B$, where $K^*$ and $V^*$ only involve the key bits and IV bits respectively. We can easily adapt our techniques to Kreyvium from Trivium. Trivium uses an 80-bit key and an 80-bit IV, while Kreyvium and TriviA-SC both use a 128-bit key and a 128-bit IV. All these ciphers have 1152 rounds. For more details of the specifications of these ciphers, we refer to [10,11,13,14].

### 4.2 The Algorithm for Estimation of Degree of Trivium-Like Ciphers

We present here an algorithm for giving an upper bound on the algebraic degree of the output of $f$ after $N$ rounds for a Trivium-like cipher, as depicted in Algorithm 2. We first initialize the degree of the initial internal state, denoted by $D^{(0)}$, then iteratively compute $D^{(t)}$ for $t = 1, 2, \cdots, N$, and finally apply numeric mapping to calculate an estimated degree for the first bit of the keystream. In Algorithm 2, we also use three sequences, denoted by $d_A$, $d_B$ and $d_C$, to record the estimated degrees of the three registers $A, B, C$. In each step of a Trivium-like cipher, three bits are updated as (4), (5) and (6). Accordingly, we compute estimated degrees for these three bits in each step $t$, denoted by $d_A^{(t)}, d_B^{(t)}$ and $d_C^{(t)}$. Then update $D^{(t)}$ from $D^{(t-1)}$. For estimating the algebraic degrees of $x_t, y_t, z_t$, we exploit two procedures `DegMul` and `DEG` for dealing with their "quadratic" and "linear" parts separately. An instance of `DegMul` is described in Algorithm 3. The other two cases are similar, and the full procedure of `DegMul` is given in Algorithm 5 in Appendix. Algorithm 3 is used to compute an upper bound on the algebraic degree of $g_A^{(t)} = z_{t-r_C} \cdot z_{t-r_C+1}$, and its correctness is shown in Lemma 4. We will demonstrate that for all $t$ with $1 \leq t \leq N$ the estimated degrees $d_A^{(t)}, d_B^{(t)}, d_C^{(t)}$ for $x_t, y_t, z_t$ are greater than or equal to their corresponding algebraic degrees, and therefore the output $\mathtt{DEG}(f, D^{(N)})$ of Algorithm 2 is a super numeric degree of the first bit of the keystream. In other words, Algorithm 2 gives an upper bound on algebraic degree of the $N$-round output bit of a Trivium-like cipher.

**Theorem 3.** *Algorithm 2 outputs a super numeric degree of the first keystream bit of an $N$-round Trivium-like cipher with $X$ as variables.*

As mentioned previously, to prove Theorem 3, it is sufficient to show the following lemma.

**Algorithm 2.** Estimation of Degree of Trivium-Like Ciphers

```
Require: Given the ANFs of the initial internal state $(A_0, B_0, C_0)$, and the set of variables $X$.
For $\lambda$ in $\{A, B, C\}$ do:
    For $t$ from $1 - n_\lambda$ to 0 do:
        $d_\lambda^{(t)} \leftarrow \deg(\lambda_0[t], X)$, where $A_0[t] = x_t$, $B_0[t] = y_t$, $C_0[t] = z_t$;
$D^{(0)} \leftarrow (d_A^{(1-n_A)}, \cdots, d_A^{(0)}, d_B^{(1-n_B)}, \cdots, d_B^{(0)}, d_C^{(1-n_C)}, \cdots, d_C^{(0)})$;
For $t$ from 1 to $N$ do:
    For $\lambda$ in $\{A, B, C\}$ do:
        $d_\lambda^{(t)} \leftarrow \max\{\texttt{DegMul}(g_\lambda^{(t)}), \texttt{DEG}(\ell_\lambda, D^{(t-1)})\}$;
    $D^{(t)} \leftarrow (d_A^{(t-n_A+1)}, \cdots, d_A^{(t)}, d_B^{(t-n_B+1)}, \cdots, d_B^{(t)}, d_C^{(t-n_C+1)}, \cdots, d_C^{(t)})$;
Return $\texttt{DEG}(f, D^{(N)})$.
```

**Algorithm 3.** $\texttt{DegMul}(g_\lambda^{(t)})$ for $\lambda = A$

```
$t_1 \leftarrow t - r_C$;
If $t_1 \leq 0$ then:
    Return $d_C^{(t_1)} + d_C^{(t_1+1)}$.
$t_2 \leftarrow t_1 - r_B$;
$d_1 \leftarrow \min\{d_B^{(t_2)} + d_C^{(t_1+1)}, d_B^{(t_2+2)} + d_C^{(t_1)}, d_B^{(t_2)} + d_B^{(t_2+1)} + d_B^{(t_2+2)}\}$;
$d_2 \leftarrow \texttt{DEG}(\ell_C, D^{(t_1)}) + d_C^{(t_1)}$;
$d_3 \leftarrow \texttt{DEG}(\ell_C, D^{(t_1-1)}) + d_C^{(t_1+1)}$;
$d \leftarrow \max\{d_1, d_2, d_3\}$;
Return $d$.
```

**Lemma 4.** *In Algorithm 2, we have* $d_A^{(t)} \geq \deg(x_t, X)$, $d_B^{(t)} \geq \deg(y_t, X)$ *and* $d_C^{(t)} \geq \deg(z_t, X)$ *for* $t \leq N$.

*Proof.* It is trivial for $t \leq 0$. Next we simply write $\deg(\cdot, X)$ as $\deg(\cdot)$. By Eqs. (4), (5) and (6), it is sufficient to prove for $1 \leq t \leq N$ that

$$d_A^{(t)} \geq \max\{\deg(z_{t-r_C} \cdot z_{t-r_C+1}), \deg(\ell_A(s^{(t-1)}))\}, \tag{7}$$

$$d_B^{(t)} \geq \max\{\deg(x_{t-r_A} \cdot x_{t-r_A+1}), \deg(\ell_B(s^{(t-1)}))\}, \tag{8}$$

and

$$d_C^{(t)} \geq \max\{\deg(y_{t-r_B} \cdot y_{t-r_B+1}), \deg(\ell_C(s^{(t-1)}))\}. \tag{9}$$

We prove them by induction. Here we provide only the details of the proof for the first inequality due to the similarity. It is clear that (7) is true for $1 \leq t \leq r_C$. Assume that (7), (8) and (9) are true for all $i \leq t - 1$. Now we prove that (7) is true for $t$ with $r_C < t \leq N$.

From Algorithm 2, we have $d_A^{(t)} \geq \texttt{DEG}(\ell_A, D^{(t-1)}) \geq \deg(\ell_A(s^{(t-1)}))$. Next we prove $d_A^{(t)} \geq \deg(z_{t-r_C} \cdot z_{t-r_C+1})$. By (6), we obtain that for $t - r_C \geq 1$,

$$z_{t-r_C} = y_{t-r_C-r_B} \cdot y_{t-r_C-r_B+1} + \ell_C(s^{(t-r_C-1)}),$$

$$z_{t-r_C+1} = y_{t-r_C-r_B+1} \cdot y_{t-r_C-r_B+2} + \ell_C(s^{(t-r_C)}),$$

and thus

$$\begin{aligned}
&z_{t-r_C} \cdot z_{t-r_C+1}\\
=&(y_{t-r_C-r_B} \cdot y_{t-r_C-r_B+1} + \ell_C(s^{(t-r_C-1)})) \cdot z_{t-r_C+1}\\
=&y_{t-r_C-r_B} \cdot y_{t-r_C-r_B+1} \cdot z_{t-r_C+1} + \ell_C(s^{(t-r_C-1)}) \cdot z_{t-r_C+1}\\
=&y_{t-r_C-r_B} \cdot y_{t-r_C-r_B+1} \cdot (y_{t-r_C-r_B+1} \cdot y_{t-r_C-r_B+2} + \ell_C(s^{(t-r_C)}))\\
&\quad + \ell_C(s^{(t-r_C-1)}) \cdot z_{t-r_C+1}\\
=&y_{t-r_C-r_B} \cdot y_{t-r_C-r_B+1} \cdot y_{t-r_C-r_B+2} + y_{t-r_C-r_B} \cdot y_{t-r_C-r_B+1} \cdot \ell_C(s^{(t-r_C)})\\
&+ \ell_C(s^{(t-r_C-1)}) \cdot z_{t-r_C+1}.
\end{aligned}$$

Denote by $Y_1, Y_2$ and $Y_3$ respectively the three summands in the above equality. By the previous assumption, we have

$$\begin{aligned}
d_C^{(t-r_C)} &\geq \deg(y_{t-r_C-r_B} \cdot y_{t-r_C-r_B+1}),\\
d_C^{(t-r_C+1)} &\geq \deg(y_{t-r_C-r_B+1} \cdot y_{t-r_C-r_B+2}),
\end{aligned}$$

and thus

$$\begin{aligned}
\deg(Y_1) \leq &\min\{\deg(y_{t-r_C-r_B}) + \deg(y_{t-r_C-r_B+1} \cdot y_{t-r_C-r_B+2}),\\
&\qquad \deg(y_{t-r_C-r_B+2}) + \deg(y_{t-r_C-r_B} \cdot y_{t-r_C-r_B+1}),\\
&\qquad \deg(y_{t-r_C-r_B}) + \deg(y_{t-r_C-r_B+1}) + \deg(y_{t-r_C-r_B+2})\}\\
\leq &\min\{\deg(y_{t-r_C-r_B}) + d_C^{(t-r_C+1)},\\
&\qquad \deg(y_{t-r_C-r_B+2}) + d_C^{(t-r_C)},\\
&\qquad \deg(y_{t-r_C-r_B}) + \deg(y_{t-r_C-r_B+1}) + \deg(y_{t-r_C-r_B+2})\}\\
\leq &\min\{d_B^{(t-r_C-r_B)} + d_C^{(t-r_C+1)},\\
&\qquad d_B^{(t-r_C-r_B+2)} + d_C^{(t-r_C)},\\
&\qquad d_B^{(t-r_C-r_B)} + d_B^{(t-r_C-r_B+1)} + d_B^{(t-r_C-r_B+2)}\} = d_1.
\end{aligned}$$

From the assumption we also have

$$\begin{aligned}
\deg(Y_2) &\leq \mathtt{DEG}(\ell_C, D^{(t-r_C)}) + d_C^{(t-r_C)} = d_2,\\
\deg(Y_3) &\leq \mathtt{DEG}(\ell_C, D^{(t-r_C-1)}) + d_C^{(t-r_C+1)} = d_3.
\end{aligned}$$

Since $\deg(z_{t-r_C} \cdot z_{t-r_C+1}) \leq \max\{\deg(Y_1), \deg(Y_2), \deg(Y_3)\} \leq \max\{d_1, d_2, d_3\}$, by Algorithms 2 and 3 we know $\deg(z_{t-r_C} \cdot z_{t-r_C+1}) \leq d_A^{(t)}$. □

**Complexity of the Algorithm.** The size of the ANF of $\ell_\lambda$ is constant and thus $\mathtt{DEG}(\ell_\lambda)$ and $\mathtt{DegMul}(g_\lambda^{(t)})$ can be calculated in constant time, for $\lambda \in \{A, B, C\}$. Therefore Algorithm 2 has time complexity of $\mathcal{O}(N)$. It requires a memory of $\mathcal{O}(N)$.

### 4.3 Experimental Results

In this section, we implement the algorithm on Trivium, Kreyvium and TriviA-SC, and reveal various upper bounds on the algebraic degrees of these ciphers. For Kreyvium, we use a modified $D^{(t)}$ in the algorithm which includes the degrees of the two extra registers (key and IV).

**When Will the Key and IV Be Sufficiently Mixed?** We take all the key and IV bits as input variables $X$, and do experiments on Trivium, Kreyvium and TriviA-SC using Algorithm 2. We list the results in Table 2. As shown in the table, Trivium does not achieve the maximum degree 160 after an initialization of 907 rounds, while Kreyvium and TriviA-SC do not achieve the maximum degree 256 after 982 rounds and 1108 rounds respectively. Though it is not an attack, this implies that Trivium behaves best among the three ciphers while TriviA-SC has a small margin towards this test of maximum algebraic degree.

**Table 2.** Lower bound on the maximum number of rounds of not achieving maximum degree for Trivium, Kreyvium and TriviA-SC with all the key and IV bits as variables ($X = (key, IV)$)

| Cipher | Trivium | Kreyvium | TriviA-SC |
|---|---|---|---|
| #Key+#IV | 160 | 256 | 256 |
| #Rounds | 907 | 982 | 1108 |

**When Will the IV Be Sufficiently Mixed?** Taking a subset of the IV as input variables and the key as parameter, the algorithm gives a chosen IV distinguisher on the cipher. Such kind of distinguishers, including cube testers, have been widely investigated on stream ciphers, *e.g.*, [3,23,24,37].

We first apply the algorithm to Trivium, Kreyvium and TriviA-SC with all the IV bits as input variables, *i.e.*, $X = IV$. In our experiments, the key is taken as parameter, that is, $\deg(k_i, X) = 0$ for any bit $k_i$ of the key. This is consistent with a distinguisher in the setting of unknown key. Our experiments show that Trivium does not achieve the maximum degree 80 after an initialization of 793 rounds, while Kreyvium and TriviA-SC do not achieve the maximum degree 128 after 862 rounds and 987 rounds respectively. We summarize our results in Table 3.

We next consider an exhaustive search on the sets of input variables $X$ which have size of around half length of the IV and contain no adjacent indexes. This is not the first time to make use of a cube that contain no adjacent indexes. Actually, the results of Aumasson *et al.* [3] and Liu *et al.* [33] have shown that we can profit from such kind of cubes in cube testers due to the non-linear structure of the update functions of Trivium. In our experiments, we set the key as parameter, and set the non-variable IV bits to be zeros. Using

**Table 3.** Lower bound on the maximum number of rounds of NOT achieving maximum degree for Trivium, Kreyvium and TriviA-SC with all the IV bits as variables ($X = IV$)

| Cipher | Trivium | Kreyvium | TriviA-SC |
|---|---|---|---|
| #IV | 80 | 128 | 128 |
| #Rounds | 793 | 862 | 987 |

Algorithm 2, we can exhaust all the cubes of size $37 \leq n \leq 40$ for Trivium, which contain no adjacent indexes, in a dozen minutes on a common PC. The amount of such cubes is $\sum_{n=37}^{40}\binom{81-n}{n} \approx 2^{25}$. Before this paper, it needs $c\sum_{n=37}^{40} 2^n\binom{81-n}{n} \approx c2^{62}$ cipher operations to test all those cubes, and the confidence of the test depends on $c$. All the cubes containing no adjacent indexes of size $61 \leq n \leq 64$ for Kreyvium and TriviA-SC are exhausted in a few hours. The amount of such cubes is $\sum_{n=61}^{64}\binom{129-n}{n} \approx 2^{30}$. By the existing methods, it needs $c\sum_{n=61}^{64} 2^n\binom{129-n}{n} \approx c2^{91}$ cipher operations to test all those cubes. The results are summarized in Table 4. The corresponding cubes are listed in Table 7 in Appendix.

As shown in Table 4, the output of 837-round Trivium has degree strictly less than 37 over a subset of IV bits with size 37, and thus the outputs of 837-round Trivium over this cube always sum to 0. Since $2^{37}$ is practical, we verify this by carrying out a test for random 100 keys. The minimum number of rounds such that the sum over this cube, *i.e.*, the superpoly of the cube, is not zero-constant is detected to be 839, which means the output of 839-round Trivium achieves the maximum degree 37 over this subset of IV bits. This shows that our lower bound on the number of attacked rounds is very sharp, and our estimation of degree is, in some ways, very close to its real value. The test also implies a distinguisher for 842-round Trivium with time complexity of around $2^{39}$, since we detect a bias of 0.46 from the 842-round output bit. We summarize

**Table 4.** Cube testers on round-reduced Trivium, Kreyvium and TriviA-SC with around half of the IV bits as variables

| Cipher | Trivium | Kreyvium | TriviA-SC (v1) | TriviA-SC (v2) | Simplified TriviA-SC |
|---|---|---|---|---|---|
| Size of cube | 37 | 61 | 63 | 62 | 63 |
| #Rounds | 837 | 872 | 1035 | 1046 | 1152 |

**Table 5.** Superpoly of round-reduced Trivium over a cube of size 37

| #Rounds | 837 | 838 | 839 | 840 | 841 | 842 |
|---|---|---|---|---|---|---|
| rate(superpoly=1) | 0 | 0 | 0.09 | 0.07 | 0.29 | 0.27 |

in Table 5 the results of the test, where the rate that the superpoly of this cube equals non-zero is given for starting from 837 rounds to 842 rounds.

As shown in Table 4, the output of 872-round Kreyvium has algebraic degree strictly less than 61 over a subset of IV bits with size 61, which implies a distinguisher on this reduced version of Kreyvium with complexity of $2^{61}$.

Our experiments also show that the output of 1035-round TriviA-SC (v1) and 1046-round TriviA-SC (v2) do not achieve maximum algebraic degree on a subset of IV bits with size 63 and size 62 respectively, which implies that we can distinguish them from random functions in $2^{63}$ and $2^{62}$ respectively. In fact, these two cubes are found much earlier before the completion of our experiments. The former is found in a second, and the latter in three minutes. By using the cube of size 63, we can also obtain a distinguisher with complexity of $2^{63}$ on the full rounds of a simplified variant of TriviA-SC (for both versions), in which the unique nonlinear term of the output function is removed.

We have also tried to search for the cubes of large size under other strategies. We exhaust all the cubes with size close to the length of the IV. Besides, we use our algorithm together with the greedy algorithm, as done in [39], to search for the best cubes of any size. Nevertheless, no better results are found.

To further evaluate the accuracy of our algorithm, we perform more experiments specially on Trivium. We compute the exact value of the algebraic degree of the output bit of reduced Trivium from 66 rounds to 426 rounds, as well as estimate the degree by our algorithm. Our experiments show that

- our estimated bound is equal to its real value for most of cases (greater than 70%), and even for the other cases their gap is only one, when taking all the key and IV bits or all the IV bits as input variables.
- our estimated bound is always equal to its real value, when taking the best cube of size 37 as input variables.

They are strong evidence of high accuracy of our algorithm. We depict in Fig. 1 our full estimation of the upper bound on the algebraic degree of reduced Trivium for the mentioned three cases. From this figure, we can see that the algebraic degree on the IV bits is almost the same as that on all the key and IV bits, and it increases much faster than that of the best cube. The former is possible due to that the key and IV bits are loaded into different registers of Trivium, and the latter due to that two adjacent variable bits accelerate the growth of the algebraic degree.

**Remarks.** The algorithm is possibly improved by further refining the estimation of the degree of $y_i \cdot y_{i+1} \cdot y_{i+2}$. However, probably because in most of cases $y_i \cdot y_{i+1} \cdot y_{i+2}$ is not dominant on the algebraic degree of $z_{i+r_B} \cdot z_{i+r_B+1}$, no improvement is found by this way in our experiments. Another possible improvement is to store the estimated degree of $y_i \cdot y_{i+1}$ and replace some $d_C^{(i+r_B)}$ with it in the procedure `DegMul`. Again, it gives no better result, at least in our experiments, probably due to that the algebraic degree of $z_{i+r_B}$ is usually equal to that of $y_i \cdot y_{i+1}$. Even though these methods show no advantages in our experiments,

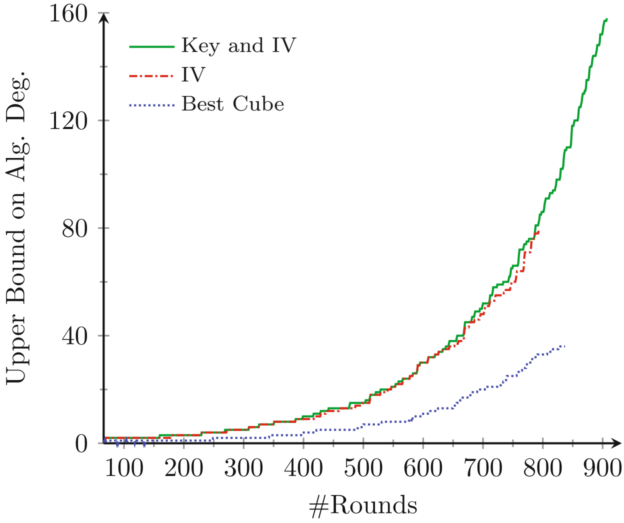

**Fig. 1.** Upper bound on the algebraic degree of reduced Trivium

they may be useful in some cases. In the following, for an instance, we will show an improved algorithm by computing the exact degrees of the internal states of the first rounds, together with the second method.

## 5 Improved Estimation of Degree of Trivium-Like Ciphers

In this section, we present an improved algorithm for estimating algebraic degree of the output of $f$ after $N$ rounds for a Trivium-like cipher, as described in Algorithm 4.

It is similar to Algorithm 2. In the improved algorithm, we compute the exact algebraic degrees of the internal states for the first $N_0$ rounds, where the degrees of $g_A^{(t)}$, $g_B^{(t)}$ and $g_C^{(t)}$ are also recorded, and use a modified $\texttt{DegMul}^*$ to replace $\texttt{DegMul}$, as depicted in Algorithm 6 in Appendix. The rest of this algorithm is the same as Algorithm 2. The output of Algorithm 4 also gives an upper bound on algebraic degree of an $N$-round Trivium-like cipher with $X$ as input variables. The replacing $\texttt{DegMul}$ with $\texttt{DegMul}^*$ does not give the improvement but guarantees the validity of the algorithm. The proof is similar to that of Algorithm 2 and thus omitted in this paper.

It is hard to assess the complexity of Algorithm 4, which depends on $N_0$ and the complexities of the ANFs of the internal states $(A_t, B_t, C_t)$ with $t \leq N_0$. It becomes much slower than Algorithm 2, as $N_0$ increases.

We apply the algorithm to Trivium, Kreyvium and TriviA-SC. It slightly improves the results in Sect. 4 for TriviA-SC, as shown in Table 6, while this is not the case for Trivium and Kreyvium. For both versions of TriviA-SC in

**Algorithm 4.** Improved Estimation of Degree of Trivium-Like Ciphers

```
Require: Given the ANFs of all internal states $(A_t, B_t, C_t)$ with $t \leq N_0$, and
  the set of variables $X$.
For $\lambda$ in $\{A, B, C\}$ do:
    For $t$ from $1 - n_\lambda$ to 0 do:
        $d_\lambda^{(t)} \leftarrow \deg(\lambda_0[t], X)$;
$D^{(0)} \leftarrow (d_A^{(1-n_A)}, \cdots, d_A^{(0)}, d_B^{(1-n_B)}, \cdots, d_B^{(0)}, d_C^{(1-n_C)}, \cdots, d_C^{(0)})$;
For $t$ from 1 to $N_0$ do:
    For $\lambda$ in $\{A, B, C\}$ do:
        $dm_\lambda^{(t)} \leftarrow \deg(g_\lambda^{(t)}, X)$;
        $d_\lambda^{(t)} \leftarrow \deg(\lambda_t[t], X)$;
    $D^{(t)} \leftarrow (d_A^{(t-n_A+1)}, \cdots, d_A^{(t)}, d_B^{(t-n_B+1)}, \cdots, d_B^{(t)}, d_C^{(t-n_C+1)}, \cdots, d_C^{(t)})$;
For $t$ from $N_0 + 1$ to $N$ do:
    For $\lambda$ in $\{A, B, C\}$ do:
        $dm_\lambda^{(t)} \leftarrow \texttt{DegMul}^*(g_\lambda^{(t)})$;
        $d_\lambda^{(t)} \leftarrow \max\{dm_\lambda^{(t)}, \texttt{DEG}(\ell_\lambda, D^{(t-1)})\}$;
    $D^{(t)} \leftarrow (d_A^{(t-n_A+1)}, \cdots, d_A^{(t)}, d_B^{(t-n_B+1)}, \cdots, d_B^{(t)}, d_C^{(t-n_C+1)}, \cdots, d_C^{(t)})$;
Return $\texttt{DEG}(f, D^{(N)})$.
```

the case $X = (key, IV)$, the number of rounds such that the output has degree less than 256 is improved from 1108 to 1121, by taking $N_0 = 340$. For TriviA-SC (v2) with $X$ being a subset of $IV$ with size of 61, the number of rounds is improved from 1032 to 1047, by taking $N_0 = 440$. This cube is listed in Table 7 in Appendix.

**Table 6.** Lower bounds on the number of rounds of NOT achieving maximum degree for TriviA-SC

| Cipher | TriviA-SC | TriviA-SC (v2) |
|---|---|---|
| $X$ | $(key, IV)$ | Subset of $IV$ |
| $\#X$ | 256 | 61 |
| #Rounds (Algorithm 2) | 1108 | 1032 |
| #Rounds (Algorithm 4) | 1121 | 1047 |

## 6 Conclusions

In this paper, we have shown a general framework of algebraic degree evaluation for NFSR-based cryptosystems. It is based on a new tool, named numeric mapping. We have also detailed the technique for efficiently finding an upper bound on the algebraic degree of Trivium-like ciphers. As illustrations, we applied it to Trivium, Kreyvium and TriviA-SC, and gained the best distinguishing attacks for all these ciphers, by an exhaustive search on a subset of the cubes that have size of around half length of the IV. To the best of our knowledge, our tool is the first theoretical one for finding an upper bound on the algebraic degree of an NFSR-based cryptosystem, and this is the first time that a cube of size beyond practical computations can be used in cryptanalysis of an NFSR-based cryptosystem. Note that cube testers are useful not only in distinguishing attacks but also in key recovery attacks. We believe that this tool is useful in both cryptanalysis and design of NFSR-based cryptosystems. In the future, it is worthy of working on its applications to key recovery attacks and to more cryptographic primitives. It is also worth a further generalization to other cryptosystems that are not built on NFSR.

**Acknowledgement.** We are grateful to Jian Guo, Wenhao Wang, and anonymous reviewers of CRYPTO 2017 for their fruitful discussions and helpful comments.

# A The Full Procedures of DegMul and DegMul*

Algorithms 5 and 6 respectively describe the full procedures of $\texttt{DegMul}(g_\lambda^{(t)})$ and $\texttt{DegMul}^*(g_\lambda^{(t)})$ for $\lambda \in \{A, B, C\}$, where $\rho(A) = C, \rho(C) = B, \rho(B) = A$.

**Algorithm 5.** $\texttt{DegMul}(g_\lambda^{(t)})$ for $\lambda \in \{A, B, C\}$

```
$t_1 \leftarrow t - r_{\rho(\lambda)}$;
If $t_1 \leq 0$ then:
    Return $d_{\rho(\lambda)}^{(t_1)} + d_{\rho(\lambda)}^{(t_1+1)}$.
$t_2 \leftarrow t_1 - r_{\rho^2(\lambda)}$;
$d_1 \leftarrow \min\{d_{\rho^2(\lambda)}^{(t_2)} + d_{\rho(\lambda)}^{(t_1+1)}, d_{\rho^2(\lambda)}^{(t_2+2)} + d_{\rho(\lambda)}^{(t_1)}, d_{\rho^2(\lambda)}^{(t_2)} + d_{\rho^2(\lambda)}^{(t_2+1)} + d_{\rho^2(\lambda)}^{(t_2+2)}\}$;
$d_2 \leftarrow \texttt{DEG}(\ell_{\rho(\lambda)}, D^{(t_1)}) + d_{\rho(\lambda)}^{(t_1)}$;
$d_3 \leftarrow \texttt{DEG}(\ell_{\rho(\lambda)}, D^{(t_1-1)}) + d_{\rho(\lambda)}^{(t_1+1)}$;
$d \leftarrow \max\{d_1, d_2, d_3\}$;
Return $d$.
```

**Algorithm 6.** $\texttt{DegMul}^*(g_\lambda^{(t)})$ for $\lambda \in \{A, B, C\}$

```
$t_1 \leftarrow t - r_{\rho(\lambda)}$;
If $t_1 \leq 0$ then:
    Return $d_{\rho(\lambda)}^{(t_1)} + d_{\rho(\lambda)}^{(t_1+1)}$.
$t_2 \leftarrow t_1 - r_{\rho^2(\lambda)}$;
$d_1 \leftarrow \min\{d_{\rho^2(\lambda)}^{(t_2)} + dm_{\rho(\lambda)}^{(t_1+1)}, d_{\rho^2(\lambda)}^{(t_2+2)} + dm_{\rho(\lambda)}^{(t_1)}, d_{\rho^2(\lambda)}^{(t_2)} + d_{\rho^2(\lambda)}^{(t_2+1)} + d_{\rho^2(\lambda)}^{(t_2+2)}\}$;
$d_2 \leftarrow \texttt{DEG}(\ell_{\rho(\lambda)}, D^{(t_1)}) + dm_{\rho(\lambda)}^{(t_1)}$;
$d_3 \leftarrow \texttt{DEG}(\ell_{\rho(\lambda)}, D^{(t_1-1)}) + d_{\rho(\lambda)}^{(t_1+1)}$;
$d \leftarrow \max\{d_1, d_2, d_3\}$;
Return $d$.
```

# B The Best Cube Testers

**Table 7.** The cubes in cube testers on round-reduced Trivium, Kreyvium and TriviA-SC with around half of the IV bits as variables

| Cipher | Cube size | Cube |
|---|---|---|
| Trivium | 37 | {0, 2, 4, 6, 8, 10, 12, 15, 17, 19, 21, 23, 25, 27, 30, 32, 34, 36, 38, 40, 42, 45, 47, 49, 51, 53, 55, 57, 60, 62, 64, 66, 68, 70, 72, 75, 79} |
| Kreyvium | 61 | {0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 29, 31, 33, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 57, 59, 61, 63, 65, 67, 69, 72, 74, 76, 78, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98, 100, 102, 104, 107, 109, 111, 113, 115, 117, 119, 122, 124, 126} |
| TriviA-SC | 61 | {0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 75, 77, 79, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99, 101, 103, 105, 107, 109, 111, 113, 115, 121, 123, 125, 127} |
| | 62 | {0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 72, 77, 79, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99, 101, 103, 105, 107, 109, 111, 113, 115, 117, 121, 123, 125, 127} |
| | 63 | {0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 72, 74, 76, 78, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99, 101, 103, 105, 107, 109, 111, 113, 115, 117, 121, 123, 125, 127} |

## References

1. Ågren, M., Hell, M., Johansson, T., Meier, W.: Grain-128a: a new version of Grain-128 with optional authentication. IJWMC **5**(1), 48–59 (2011)
2. Aumasson, J., Dinur, I., Henzen, L., Meier, W., Shamir, A.: Efficient FPGA implementations of high-dimensional cube testers on the stream cipher Grain-128. IACR Cryptology ePrint Archive **2009**, p. 218 (2009)
3. Aumasson, J.-P., Dinur, I., Meier, W., Shamir, A.: Cube testers and key recovery attacks on reduced-round MD6 and trivium. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 1–22. Springer, Heidelberg (2009). doi:10.1007/978-3-642-03317-9_1
4. Aumasson, J.-P., Henzen, L., Meier, W., Naya-Plasencia, M.: Quark: A lightweight hash. In: Mangard, S., Standaert, F.-X. (eds.) CHES 2010. LNCS, vol. 6225, pp. 1–15. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15031-9_1
5. Aumasson, J., Henzen, L., Meier, W., Naya-Plasencia, M.: Quark: A lightweight hash. J. Cryptol. **26**(2), 313–339 (2013)

6. Boura, C., Canteaut, A.: On the influence of the algebraic degree of $F^{-1}$ on the algebraic degree of $G \circ F$. IEEE Trans. Inf. Theor. **59**(1), 691–702 (2013)
7. Boura, C., Canteaut, A., 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). doi:10.1007/978-3-642-21702-9_15
8. Cannière, C.: Trivium: A stream cipher construction inspired by block cipher design principles. In: Katsikas, S.K., López, J., Backes, M., Gritzalis, S., Preneel, B. (eds.) ISC 2006. LNCS, vol. 4176, pp. 171–186. Springer, Heidelberg (2006). doi:10.1007/11836810_13
9. Cannière, C., Dunkelman, O., Knežević, M.: KATAN and KTANTAN—A family of small and efficient hardware-oriented block ciphers. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 272–288. Springer, Heidelberg (2009). doi:10.1007/978-3-642-04138-9_20
10. Cannière, C., Preneel, B.: Trivium. In: Robshaw, M., Billet, O. (eds.) New Stream Cipher Designs. LNCS, vol. 4986, pp. 244–266. Springer, Heidelberg (2008). doi:10.1007/978-3-540-68351-3_18
11. Canteaut, A., Carpov, S., Fontaine, C., Lepoint, T., Naya-Plasencia, M., Paillier, P., Sirdey, R.: 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). doi:10.1007/978-3-662-52993-5_16
12. 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). doi:10.1007/3-540-46035-7_34
13. 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). doi:10.1007/978-3-662-48324-4_17
14. Chakraborti, A., Nandi, M.: TriviA-ck-v2. CAESAR Submission (2015). http://competitions.cr.yp.to/round2/triviackv2.pdf
15. 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). doi:10.1007/978-3-540-45146-4_11
16. Courtois, N., Klimov, A., Patarin, J., Shamir, A.: Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 392–407. Springer, Heidelberg (2000). doi:10.1007/3-540-45539-6_27
17. Courtois, N.T., Meier, W.: Algebraic attacks on stream ciphers with linear feedback. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 345–359. Springer, Heidelberg (2003). doi:10.1007/3-540-39200-9_21
18. 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). doi:10.1007/3-540-36178-2_17
19. 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). doi:10.1007/978-3-642-25385-0_18
20. 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). doi:10.1007/978-3-662-46800-5_28

21. 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). doi:10.1007/978-3-642-01001-9_16
22. 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). doi:10.1007/978-3-642-21702-9_10
23. Englund, H., Johansson, T., Sönmez Turan, M.: A framework for chosen IV statistical analysis of stream ciphers. In: Srinathan, K., Rangan, C.P., Yung, M. (eds.) INDOCRYPT 2007. LNCS, vol. 4859, pp. 268–281. Springer, Heidelberg (2007). doi:10.1007/978-3-540-77026-8_20
24. Fischer, S., Khazaei, S., Meier, W.: Chosen IV statistical analysis for key recovery attacks on stream ciphers. In: Vaudenay, S. (ed.) AFRICACRYPT 2008. LNCS, vol. 5023, pp. 236–245. Springer, Heidelberg (2008). doi:10.1007/978-3-540-68164-9_16
25. Fontaine, C.: Nonlinear feedback shift register. In: van Tilborg, H.C.A., Jajodia, S. (eds.) Encyclopedia of Cryptography and Security, 2nd edn, pp. 846–848. Springer, US (2011)
26. 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). doi:10.1007/978-3-662-43933-3_26
27. Hell, M., Johansson, T., Maximov, A., Meier, W.: A stream cipher proposal: Grain-128. In: 2006 IEEE International Symposium on Information Theory, pp. 1614–1618. IEEE (2006)
28. 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). doi:10.1007/978-3-540-68351-3_14
29. Knellwolf, S., Meier, W., Naya-Plasencia, M.: Conditional differential cryptanalysis of trivium and KATAN. In: Miri, A., Vaudenay, S. (eds.) SAC 2011. LNCS, vol. 7118, pp. 200–212. Springer, Heidelberg (2012). doi:10.1007/978-3-642-28496-0_12
30. Knudsen, L.R.: Truncated and higher order differentials. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 196–211. Springer, Heidelberg (1995). doi:10.1007/3-540-60590-8_16
31. Knudsen, L., Wagner, D.: Integral cryptanalysis. In: Daemen, J., Rijmen, V. (eds.) FSE 2002. LNCS, vol. 2365, pp. 112–127. Springer, Heidelberg (2002). doi:10.1007/3-540-45661-9_9
32. Lai, X.: Higher order derivatives and differential cryptanalysis. In: Proceeding Symposium Communication and Coding Cryptography. Kluwer Academic Publishers, pp. 227–233 (1994)
33. Liu, M., Lin, D., Wang, W.: Searching cubes for testing Boolean functions and its application to Trivium. In: IEEE International Symposium on Information Theory (ISIT 2015), Hong Kong, China, 14–19 June 2015, pp. 496–500. IEEE (2015)
34. Maximov, A., Biryukov, A.: Two trivial attacks on Trivium. In: Adams, C., Miri, A., Wiener, M. (eds.) SAC 2007. LNCS, vol. 4876, pp. 36–55. Springer, Heidelberg (2007). doi:10.1007/978-3-540-77360-3_3
35. Moriai, S., Shimoyama, T., Kaneko, T.: Higher order differential attak of CAST cipher. In: Vaudenay, S. (ed.) FSE 1998. LNCS, vol. 1372, pp. 17–31. Springer, Heidelberg (1998)
36. Oswald, E., Fischlin, M. (eds.): EUROCRYPT 2015. LNCS, vol. 9056. Springer, Heidelberg (2015)

37. Saarinen, M.O.: Chosen-IV statistical attacks on estream ciphers. In: Malek, M., Fernández-Medina, E., Hernando, J. (eds.): SECRYPT 2006, Proceedings of the International Conference on Security and Cryptography, Setúbal, Portugal, 7–10 August 2006, SECRYPT is part of ICETE - The International Joint Conference on e-Business and Telecommunications, pp. 260–266. INSTICC Press (2006)
38. Sarkar, S., Maitra, S., Baksi, A.: Observing biases in the state: case studies with Trivium and trivia-sc. Des. Codes Crypt. **82**(1–2), 351–375 (2017)
39. Stankovski, P.: Greedy distinguishers and nonrandomness detectors. In: Gong, G., Gupta, K.C. (eds.) INDOCRYPT 2010. LNCS, vol. 6498, pp. 210–226. Springer, Heidelberg (2010). doi:10.1007/978-3-642-17401-8_16
40. Todo, Y.: Structural evaluation by generalized integral property, pp. 287–314. [36]
41. Todo, Y., Isobe, T., Hao, Y., Meier, W.: Cube attacks on non-blackbox polynomials based on division property. In: Proceedings of 37th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO 2017), Santa Barbara, CA, USA, 20–24 August 2017 (2017)
42. Todo, Y., Morii, M.: Bit-based division property and application to Simon family. In: Peyrin, T. (ed.) FSE 2016. LNCS, vol. 9783, pp. 357–377. Springer, Heidelberg (2016). doi:10.1007/978-3-662-52993-5_18
43. Vardasbi, A., Salmasizadeh, M., Mohajeri, J.: Superpoly algebraic normal form monomial test on Trivium. IET Inform. Secur. **7**(3), 230–238 (2013)
44. Wu, H.: ACORN: a lightweight authenticated cipher (v3). CAESAR Submission, (2016). http://competitions.cr.yp.to/round3/acornv3.pdf
45. Xu, C., Zhang, B., Feng, D.: Linear cryptanalysis of FASER128/256 and TriviA-ck. In: Meier, W., Mukhopadhyay, D. (eds.) INDOCRYPT 2014. LNCS, vol. 8885, pp. 237–254. Springer, Cham (2014). doi:10.1007/978-3-319-13039-2_14