On the Field-Based Division Property: Applications to MiMC, Feistel MiMC and GMiMC

Jiamin Cui[1,3], Kai Hu[2], Meiqin Wang[1,3,4(✉)], and Puwen Wei[1,3]

[1] School of Cyber Science and Technology, Shandong University, Qingdao, Shandong, China. cuijiamin@mail.sdu.edu.cn,mqwang@sdu.edu.cn,pwei@sdu.edu.cn
[2] School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore. kai.hu@ntu.edu.sg
[3] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Qingdao, Shandong, China.
[4] Quan Cheng Shandong Laboratory.

Abstract. Recent practical applications using advanced cryptographic protocols such as multi-party computations (MPC) and zero-knowledge proofs (ZKP) have prompted a range of novel symmetric primitives described over large finite fields, characterized as arithmetization-oriented (AO) ciphers. Such designs, aiming to minimize the number of multiplications over fields, have a high risk of being vulnerable to algebraic attacks, especially to the higher-order differential attack. Thus, it is significant to carefully evaluate the growth of their algebraic degree. However, the degree estimation for AO ciphers has been a challenge for cryptanalysts due to the lack of general and accurate methods.

In this paper, we extend the division property, a state-of-the-art framework for finding the upper bound of the algebraic degree over binary fields, to the scope of $\mathbb{F}_{2^n}$. It is a generic method to detect the algebraic degree for AO ciphers, even applicable to Feistel ciphers which have no better bounds than the trivial exponential one. In this general division property, our idea is to evaluate whether the polynomial representation of a block cipher contains some specific monomials. With a deep investigation of the arithmetical feature, we introduce the propagation rules of monomials for field-based operations, which can be efficiently modeled using the bit-vector theory of SMT. Then the new searching tool for degree estimation can be constructed due to the relationship between the algebraic degree and the exponents of monomials.

We apply our new framework to some important AO ciphers, including Feistel MiMC, GMiMC, and MiMC. For Feistel MiMC, we show that the algebraic degree grows significantly slower than the native exponential bound. For the first time, we present a secret-key higher-order differential distinguisher for up to 124 rounds, much better than the 83-round distinguisher for Feistel MiMC permutation proposed at CRYPTO 2020. We also exhibit a full-round zero-sum distinguisher with a data complexity of 2^{251}. Our method can be further extended for the general Feistel structure with more branches and exhibit higher-order differential distinguishers against the practical instance of GMiMC for up to 50 rounds.

For MiMC in SP-networks, our results correspond to the exact algebraic degree proved by Bouvier et al. We also point out that the number of rounds in MiMC's specification is not sufficient to guarantee the security against the higher-order differential attack for MiMC-like schemes with different exponents. The investigation of different exponents provides some guidance on the cipher design.

Keywords: Degree Evaluation, Division Property, Finite Field, MiMC, Feistel Network

1 Introduction

The recent progress of advanced cryptographic protocols such as multi-party computations (MPC) and zero-knowledge proofs (ZKP) has motivated new insights into the design paradigm. These innovative primitives, characterized as *arithmetization-oriented* (AO) *ciphers*, focus more on the arithmetic metrics. In the case of MPC-friendly constructions, the goal is to minimize the number of multiplications in large finite fields. Examples include MiMC [3] and its generalizations Feistel MiMC and GMiMC [3,2], HadesMiMC [24], Vision and Rescue [4] and Ciminion [21].

AO cipher designs are quite different from the traditional ones. Instead of symmetric primitives whose non-linear layers are usually composed of relatively small S-boxes (typically 4 or 8 bits), AO ciphers tend to use the non-linear function with an explicit and compact algebraic representation over large finite fields (e.g., power maps like $x \mapsto x^d$ for some odd integer d). Statistical attacks such as differential [9] and linear cryptanalysis [31], which are two of the most powerful classical cryptanalytic tools, appear not to threaten the security of these new primitives. Consequently, algebraic attacks, especially the higher-order differential attack [29], usually determine their overall security level. As a concrete example, Eichlseder et al. proposed a new upper bound on the algebraic degree for low-degree key-alternating ciphers over $\mathbb{F}_{2^n}$ [22], based on which they successfully mounted a key-recovery attack on full-round MiMC. Fairly speaking, the algebraic degree is the most crucial security property of AO ciphers. It is of great importance to devise new tools for their degree estimations.

Related work. Different methods and tools for degree evaluation have always been an important topic in the literature. Trivially, the algebraic degree of the composition of two functions F and G is bounded by $\deg(F \circ G) \leq \deg(F) \cdot \deg(G)$. However, if iterated, the resulting exponential bound fails to show the real growth of the algebraic degree for many cryptographic primitives, especially after a high number of rounds. The first improvement of the trivial bound was proposed by Canteaut and Videau at EUROCRYPT 2002 [15]. Later, Boura et al. focused on the iterated SPN schemes over $\mathbb{F}_{2^n}^t$ and presented a tighter upper bound [12]. Subsequently, more improved upper bounds for SPN schemes were

proposed through comprehensive consideration of the underlying building blocks. By further studying the influence of the algebraic degree of F^{-1}, Boura and Canteaut [10] proposed a tighter bound than [12]. Recently at FSE 2022 [18], the influence of the linear layer on the algebraic degree was also noticed and the current best bounds for SPN schemes with large and low-degree S-boxes over $\mathbb{F}_{2^n}^t$ were presented. Moreover, for Even-Mansour schemes, a special case of SPN schemes, Eichlseder et al. pointed out that the algebraic degree grows linearly with the number of rounds [22] for ciphers with low-degree round functions. As an application, they managed to give a higher-order differential distinguisher on almost full MiMC. Very recently, by carefully tracing the evolution of the exponents, Bouvier et al. presented a tighter bound for ciphers based on iterated power functions [14], leading to the exact algebraic degree estimation for MiMC. However, there is no improved bound for Feistel schemes except the trivial bound. Consequently, although the general method is more universal, if we are not able to exploit the information of the components in a more fine-grained way, the resulting algebraic degree will not be accurate enough.

Besides the above-mentioned methods, another approach for degree estimation is based on division property, a state-of-the-art framework for finding integral property proposed by Todo at EUROCRYPT 2015 [34]. It is currently the optimal way to estimate the algebraic degree in terms of accuracy as pointed out in [17]. The division property was initially word-oriented and then extended to bit level [35], referred to as the bit-based division property and three-subset bit-based division property [35]. Subsequently, there was a lot of research focusing on this topic to explain the imperfect nature inherent or extend the application scope with the help of automatic approaches [11,38,33,37,13,30,26,20]. At EUROCRYPT 2020, Hao et al. proposed the three-subset bit-based division property without unknown subset (3SDPwoU) [25] and achieves perfect accuracy. The monomial prediction proposed by Hu et al. [27] is another language of division property from a complete polynomial viewpoint. It allows us to precisely determine whether or not a specific monomial appears in the ANF. Besides, they also provide a framework to detect the integral properties more precisely than but with similar efficiency as the two-subset division property for block ciphers. Throughout this paper, we use the division porperty and monomial prediction to denote the same technology without making strict distinctions. Despite of their powerfulness, the division property/monomial prediction requires the ANF of local components, which is too complicated to be calculated or stored in practice for large finite fields. Even if we know the ANF, the existing tools cannot handle the modeling for S-boxes with a size larger than 32 bits [36] in practical time to the best of our knowledge. Overall, the bit-based division property fails to be useful for `AO` ciphers. However, `AO` ciphers can be directly regarded as multivariate polynomials over public variables (e.g., plaintext variables) and secret variables (e.g., key variables) in $\mathbb{F}_{2^n}$. This inspires us to focus on the algebraic essentials of division property and thus take benefit from the concise polynomial representations over fields.

1.1 Our Contribution

In this paper, we extend the division property, a state-of-the-art method for finding integral properties over binary fields, to the scope of the binary extension field $\mathbb{F}_{2^n}$, called *general monomial prediction* (GMP). It is a generic method to evaluate the algebraic degree for ciphers over fields, in the way of studying whether or not the polynomial representation of a block cipher described over $\mathbb{F}_{2^n}$ contains some specific monomials by decomposing the cipher into a sequence of simpler functions and tracing the monomial propagations. We then propose the propagation rules of the monomials based on the arithmetical features and model them with the aid of the bit-vector theory of Satisfiability Modulo Theories (SMT). Finally, by tracing the evolution of exponents for the monomials, we construct an SMT-based searching tool for the degree estimation of ciphers over $\mathbb{F}_{2^n}$. We apply our algorithm to some important arithmetization-oriented block ciphers, including MiMC, Feistel MiMC, GMiMC, and their variants. The full version of the paper can be found in [1]. All the source codes are available at https://github.com/iljido/GeneralMonomialPrediction.

- For Feistel MiMC, we show in particular that its algebraic degree grows obviously slower than the originally believed one. More precisely, after an initial linear growth, the algebraic degree grows rather slow for a long period, along with several large plateaus until reaching the maximal degree. While the previous work only handles the permutation Feistel MiMC, using our results, for the first time we present a secret-key higher-order differential distinguisher covering a total of 124 rounds. It is 41 rounds more than the previous best distinguisher of permutation Feistel MiMC. We also establish a known-key zero-sum distinguisher for the full-round Feistel MiMC over $\mathbb{F}_{2^n}$ with a data complexity of 2^{251}. Our method can be extended to more branches and we successfully find the currently longest secret-key higher-order differential distinguisher for practical instance of block cipher GMiMC reaching up to 50 rounds, 10 rounds longer than the previous best distinguisher.
- We also investigate the algebraic degree of MiMC-like schemes with generic exponents d. For exponents of the form $d = 2^l - 1$, we extend the higher-order differential distinguishers by one or two more rounds for different instances compared to the currently best results in [22]. For exponents of the form $d = 2^l + 1$, we find distinguishers with lower data complexities for $d = 5, 9, 17$. Our results for MiMC with $d = 3$ are consistent with the exact algebraic degree proved in [14]. Based on our results, we point out that the formula for the number of rounds used in MiMC specification [3] is not sufficient to guarantee security against the higher-order differential attack. This investigation of different exponents provides some guidance on the design.
- Moreover, we present a comprehensive analysis of the degree growth of MiMC-like schemes in (unbalanced) Feistel networks and prove a theoretical upper bound that improves the trivial exponential bound.

All the results are summarized in Table 1, Table 2 and Table 3. Our experiments are implemented in the AMD EPYC 7302 CPU @ 3.0 GHz with 8 threads.

Table 1: Higher-order differential distinguishers for $\mathsf{FeistelMiMC}_3(129, r)$.

Security	#Rounds	Target		Attack		Time	Source
		Permutation	Block Cipher	#Rounds	Cost		
129	164	✓	–	82	$2^{127\dagger}$	–	[8]
		✓	✓	**82**	$\mathbf{2^{127}}$	< 1 min	**Sec 5.1**
258	166	✓	–	83	2^{129}	–	[8]
		✓	✓	**83**	$\mathbf{2^{129}}$	< 1 min	**Sec 5.1**
		✓	✓	**124**	$\mathbf{2^{257}}$	< 5 min	**Sec 5.1**

† This complexity is calculated using the formula in [8] with subgroup of size 2^{127}.

Table 2: Zero-sum distinguishers for $\mathsf{FeistelMiMC}_3(129, r)$.

Security	#Rounds	Attack		Source
		#Rounds	Cost	
129	164	162	$2^{127\dagger}$	[8]
		163	$\mathbf{2^{127}}$	**Sec 5.1**
258	166	164	2^{129}	[8]
		165	$\mathbf{2^{129}}$	**Sec 5.1**
		166	$\mathbf{2^{251}}$	**Sec 5.1**

† This complexity is calculated using the formula in [8] with subgroup of size 2^{127}.

1.2 Outline

The rest of this paper is organized as follows. In Section 2, we revisit some background knowledge about polynomial representations, the monomial prediction, and SMT solvers. In Section 3, we propose the principle of *general monomial prediction* and present the new searching model for degree estimation. For a better insight into the degree estimation, we prove a theoretical upper bound on the algebraic degree for ciphers in (unbalanced) Feistel-networks with low-degree round functions in Section 4. Section 5 shows the applications to MiMC, Feistel MiMC, and GMiMC. We conclude the paper in Section 6.

2 Preliminaries

2.1 Notations

Let $\mathbb{F}_2^n$ denote the n-dimensional vector space over the finite field $\mathbb{F}_2$. $\mathbb{F}_{2^n}^t$ denotes the t-fold Cartesian product of the binary extension field $\mathbb{F}_{2^n}$. For any n-bit vector $\boldsymbol{u} = (u[0], \cdots, u[n-1]) \in \mathbb{F}_2^n$, the Hamming weight of $\boldsymbol{u}$ is $wt(\boldsymbol{u}) = \sum_{i=0}^{n-1} u[i]$. For any $a \in \mathbb{F}_{2^n}$, we have $a = \sum_{i=0}^{n-1} a[i] \cdot 2^i$ for $a[i] \in \{0, 1\}$ and $wt(a) = \sum_{i=0}^{n-1} a[i]$. For any $a, a' \in \mathbb{F}_{2^n}$, we define $a \preceq a'$ if $a[i] \leq a'[i]$ for all

Table 3: Distinguishers for different instances of $\mathsf{MiMC}_d(129, r)$.

	d/l	$n \cdot \log_d(2)$	Attack		Source
			#Rounds	Cost	
$d = 2^l - 1$	7/3	46	45	2^{127}	[22]
			45	$\mathbf{2^{124}}$	**Section 5.2**
			46	$\mathbf{2^{127}}$	**Section 5.2**
	15/4	34	32	2^{126}	[22]
			32	$\mathbf{2^{125}}$	**Section 5.2**
			33	$\mathbf{2^{125}}$	**Section 5.2**
	31/5	27	25	2^{124}	[22]
			25	$\mathbf{2^{121}}$	**Section 5.2**
			27	$\mathbf{2^{128}}$	**Section 5.2**
$d = 2^l + 1$	3/1	82	80	2^{128}	[22]
			81	2^{127}	[14]
			81	$\mathbf{2^{127}}$	**Section 5.2**
	5/2	56	54	2^{125}	[14]
			54	$\mathbf{2^{124}}$	**Section 5.2**
			55	2^{128}	[22]
			55	2^{127}	[14]
			55	$\mathbf{2^{127}}$	**Section 5.2**
	9/3	41	40	2^{127}	[22]
			40	2^{125}	[14]
			40	$\mathbf{2^{124}}$	**Section 5.2**
			41	2^{128}	[14]
			41	$\mathbf{2^{127}}$	**Section 5.2**
	17/4	32	31	2^{127}	[22]
			32	2^{128}	[14]
			32	$\mathbf{2^{127}}$	**Section 5.2**

i, $a \succeq a'$ if $a[i] \geq a'[i]$ for all i. We use $\oplus$ as addition over $\mathbb{F}_2$ or $\mathbb{F}_{2^n}$. 0^n or 1^n represents the all-zeros or all-ones vector of length n, respectively.

Polynomial representations. Let $F\colon \mathbb{F}_{2^n}^t \to \mathbb{F}_{2^n}$ be a function over $\mathbb{F}_{2^n}[x_0, x_1, \cdots, x_{t-1}]/\left\langle x_0^{2^n} - x_0, x_1^{2^n} - x_1, \cdots, x_{t-1}^{2^n} - x_{t-1}\right\rangle$. F can be uniquely expressed by a polynomial over $\mathbb{F}_{2^n}$ with t variables $x_0, x_1, \cdots, x_{t-1} \in \mathbb{F}_{2^n}$, as

$$F(x_0, \cdots, x_{t-1}) = \sum_{\boldsymbol{v}=(v_0,\cdots,v_{t-1})\in\{0,1,\cdots,2^n-1\}^t} \varphi(\boldsymbol{v}) \cdot \pi_{\boldsymbol{v}}(\boldsymbol{x}) \tag{1}$$

where the coefficient $\varphi(\boldsymbol{v}) \in \mathbb{F}_{2^n}$. We call the degree of a single variable in F as *univariate degree* and the degree of F as a multivariate polynomial as *multivariate degree*. The maximum univariate degree is $2^n - 1$. When $t = 1$,

the maximum univariate degree is $2^n - 2$ if F is invertible since the maximal algebraic degree of invertible functions over $\mathbb{F}_{2^n}$ is $n - 1$.

In Equation (1), $\pi_{\boldsymbol{v}}(\boldsymbol{x}) = \prod_{i=0}^{t-1} x_i^{v_i} = x_0^{v_0} \cdot \ldots \cdot x_{t-1}^{v_{t-1}}$ is called a monomial over $\mathbb{F}_{2^n}$. If the coefficient of $\pi_{\boldsymbol{v}}(\boldsymbol{x})$ in F is a constant $c \neq 0$, we say $\pi_{\boldsymbol{v}}(\boldsymbol{x})$ is contained by F, denoted by $\pi_{\boldsymbol{v}}(\boldsymbol{x}) \to F$. Otherwise, if the coefficient of $\pi_{\boldsymbol{v}}(\boldsymbol{x})$ in F is 0, $\pi_{\boldsymbol{v}}(\boldsymbol{x})$ is not contained by F, denoted by $\pi_{\boldsymbol{v}}(\boldsymbol{x}) \nrightarrow F$.

As is well-known, the function F can as well be represented at bit level with $N = n \cdot t$ variables. The i-th output element is defined by the coordinate function

$$F_i(y_0, \cdots, y_{N-1}) = \sum_{\boldsymbol{u}=(u_0,\cdots,u_{N-1}) \in \{0,1\}^N} \rho_i(\boldsymbol{u}) \cdot \pi_{\boldsymbol{u}}(\boldsymbol{y}). \tag{2}$$

The coefficient $\rho_i(\boldsymbol{u}) \in \mathbb{F}_2$ can be computed by the *Möbius transform*. $\pi_{\boldsymbol{u}}(\boldsymbol{y}) = \prod_{i=0}^{N-1} y_i^{u_i} = y_0^{u_0} \cdot \ldots \cdot y_{N-1}^{u_{N-1}}$ is called a monomial. If the coefficient of $\pi_{\boldsymbol{u}}(\boldsymbol{y})$ in F_i is 1, we say $\pi_{\boldsymbol{u}}(\boldsymbol{y})$ is contained by F_i, denoted by $\pi_{\boldsymbol{u}}(\boldsymbol{y}) \to F_i$. Otherwise, $\pi_{\boldsymbol{u}}(\boldsymbol{y})$ is not contained by F_i, denoted by $\pi_{\boldsymbol{u}}(\boldsymbol{y}) \nrightarrow F_i$.

This representation is also referred to as *algebraic normal form* (ANF) of Boolean functions. Essentially, we can see that the polynomial representation and ANF of F are equivalent when $n = 1$.

Definition 1 (ANF and Algebraic Degree). *Let $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$ be a Boolean function. Its algebraic normal form (ANF) is given as*

$$f(\boldsymbol{x}) = f(x[0], x[1], \cdots, x[n-1]) = \bigoplus_{\boldsymbol{u} \in \mathbb{F}_2^n} \rho(\boldsymbol{u}) \cdot \boldsymbol{x}^{\boldsymbol{u}} \tag{3}$$

where the coefficient $\rho(\boldsymbol{u}) \in \mathbb{F}_2$ and $\boldsymbol{x}^{\boldsymbol{u}} = \prod_{i=0}^{n-1} x[i]^{u[i]}$. Then the algebraic degree of f is defined as

$$\delta(f) = \max\{wt(\boldsymbol{u}) \mid \boldsymbol{u} \in \mathbb{F}_{2^n}, \rho(\boldsymbol{u}) \neq 0\}.$$

If $\boldsymbol{f}\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ is a vectorial Boolean function, then the algebraic degree is defined as the maximal algebraic degree of its coordinate functions f_i, i.e., $\delta(\boldsymbol{f}) = \max\{\delta(f_i) \mid 0 \leq i < m\}$.

The link between the algebraic degree and the univariate degree of a vectorial Boolean function is well-known.

Proposition 1 ([16]). *For any univariate function $F\colon \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ as*

$$F(x) = \sum_{v \in \{0,1,\cdots,2^n-1\}} \varphi(v) \cdot x^v,$$

the algebraic degree of F as a vectorial Boolean function is the maximum Hamming weight of the exponents for the non-vanishing monomials, i.e.,

$$\delta(F) = \max_{0 \leq v \leq 2^n-1} \{wt(v) \mid \varphi(v) \neq 0\}.$$

Corollary 1. *For $x_0, x_1, \cdots, x_{t-1} \in \mathbb{F}_{2^n}$, the algebraic degree of a monomial $\pi_{\boldsymbol{u}}(\boldsymbol{x}) = x_0^{u_0} \cdot \ldots \cdot x_{t-1}^{u_{t-1}}$ is given by $\sum_{i=0}^{t-1} wt(u_i)$.*

2.2 Monomial Prediction

In this paper, we mainly take the framework of the monomial prediction to simplify the exposition. The monomial prediction, proposed by Hu et al. in [27], is another language of division property from a pure algebraic perspective. By counting the so-called monomial trails, the monomial prediction can determine if a monomial of the plaintext or IV appears in the polynomial of the output of the cipher, proved to be equivalent to the three-subset bit-based division property without unknown subsets [25].

Let $\boldsymbol{f}\colon \mathbb{F}_2^{n_0} \to \mathbb{F}_2^{n_r}$ be a composite vectorial Boolean function of a sequence of smaller functions $\boldsymbol{f}^{(i)}\colon \mathbb{F}_2^{n_i} \to \mathbb{F}_2^{n_{i+1}}$, $0 \le i \le r-1$, as

$$\boldsymbol{f} = \boldsymbol{f}^{(r-1)} \circ \boldsymbol{f}^{(r-2)} \circ \cdots \circ \boldsymbol{f}^{(0)}$$

where $\boldsymbol{x}^{(i+1)} = \boldsymbol{f}^{(i)}(\boldsymbol{x}^{(i)})$. Considering the function $\boldsymbol{f}^{(i)}$, if the ANF of $\boldsymbol{f}^{(i)}$ is available, we can find the monomial $\pi_{\boldsymbol{u}^{(i+1)}}(\boldsymbol{x}^{(i+1)})$ that contains the monomial $\pi_{\boldsymbol{u}^{(i)}}(\boldsymbol{x}^{(i)})$ for any $\boldsymbol{u}^{(i)}$ easily, denoted by $\pi_{\boldsymbol{u}^{(i)}}(\boldsymbol{x}^{(i)}) \to \pi_{\boldsymbol{u}^{(i+1)}}(\boldsymbol{x}^{(i+1)})$. If we can find an r-round transition connecting $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)})$ and $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ as

$$\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \to \pi_{\boldsymbol{u}^{(1)}}(\boldsymbol{x}^{(1)}) \to \cdots \to \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)}),$$

then the r-round transition is denoted by $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \rightsquigarrow \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$, called a *monomial trail*. The set of all monomial trails from $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)})$ to $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ are denoted by $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \bowtie \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$. The size of the monomial trails determines whether $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \to \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$. If there is no trail from $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)})$ to $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$, we say $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \not\rightsquigarrow \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ and hence $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \nrightarrow \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$.

2.3 SMT Solvers

A recent approach to construct automatic tools is to formulate the searching problems into some mathematical problems and delegate the solving task to the powerful off-the-shelf solvers. The Satisfiability Modulo Theories (SMT) [6] is a problem of determining whether logical formulas in the first-order logic is satisfiable. It is a generalization of the Boolean Satisfiability Problem (SAT) [19]. SMT formulas provide much richer modeling language than SAT formulas such as bit-vectors, which give more flexibility in the interpretation of mathematical problems.

A bit-vector variable is a string of Boolean variables that can represent either a bit-vector or an integer. The set of the basic bit-vector operations is a combination of arithmetic operations and bit-wise operations. We list the operations used in the following sections in Table 4.

There are many public available solvers to solve SMT problems. We construct our model using the CVC [7] input language and take STP [23] and Cryptominisat5 [32] as our solvers in the paper. For more details about STP and CVC, readers are encouraged to refer to http://stp.github.io/.

Table 4: Basic bit-vector operations.

$x \wedge y$	bit-wise AND of x and y	$x + y$	addition of x and y
$x \vee y$	bit-wise OR of x and y	$x \times y$	multiplition of x and y
$x \oplus y$	bit-wise XOR of x and y	$x = y$	x is equal to y
$x \mid\mid y$	concatenation of x and y	$x \neq y$	x is not equal to y
$x \ll i$	x left shift by i bits	$x \leq y$	x is less than or equal to y
$x \gg i$	x right shift by i bits	$x \geq y$	x is greater than or equal to y

3 General Monomial Prediction

Let $\boldsymbol{y} = \boldsymbol{F}(\boldsymbol{x})$ be a function from $\mathbb{F}_{2^n}^t$ to $\mathbb{F}_{2^n}^s$. We focus on the exponents of $\boldsymbol{x}$ so that the algebraic degree can be estimated based on its relationship with the Hamming weight of exponents. In Section 3.1, we will introduce how to trace the transition of the exponents by generalizing the monomial prediction from $\mathbb{F}_2$ to the finite field $\mathbb{F}_{2^n}$, referred to as *general monomial prediction* (GMP). Since any function F can be represented as a sequence of basic operations such as XOR, AND, COPY, m-COPY, and POWER, we give the propagation rules for these basic functions by investigating the arithmetical features in Section 3.2 and provide their SMT models in Section 3.3. Finally in Section 3.4, by setting the initial constraints and stopping rules appropriately, the problem of degree estimation for ciphers over fields can be converted into an SMT problem and solved efficiently.

3.1 Definition of General Monomial Prediction

Let $\boldsymbol{y} = \boldsymbol{F}(\boldsymbol{x})$ be a function from $\mathbb{F}_{2^n}^t$ to $\mathbb{F}_{2^n}^s$, where $\boldsymbol{x} = (x_0, \cdots, x_{t-1})$ and $\boldsymbol{y} = (y_0, \cdots, y_{s-1})$. By *general monomial prediction* we mean the problem of whether a particular monomial $\boldsymbol{y}^{\boldsymbol{v}}$ is contained by $\boldsymbol{x}^{\boldsymbol{u}}$, denoted by $\boldsymbol{x}^{\boldsymbol{u}} \to \boldsymbol{y}^{\boldsymbol{v}}$. Notice that we make no distinction between the secret variables and public variables here and they are all treated as symbolic variables. While it is a trivial problem if the polynomial representation of $\boldsymbol{F}$ is available, $\boldsymbol{F}$ is usually too complicated to be computed or stored in practice for most symmetric primitives and we are limited to knowing the local components of $\boldsymbol{F}$.

Let $\boldsymbol{F}\colon \mathbb{F}_{2^n}^{t_0} \to \mathbb{F}_{2^n}^{t_r}$ be a composite function over $\mathbb{F}_{2^n}$ consisting of a sequence of smaller functions $\boldsymbol{F}^{(i)}\colon \mathbb{F}_{2^n}^{t_i} \to \mathbb{F}_{2^n}^{t_{i+1}}$, $0 \leq i \leq r-1$, as

$$\boldsymbol{F} = \boldsymbol{F}^{(r-1)} \circ \boldsymbol{F}^{(r-2)} \circ \cdots \circ \boldsymbol{F}^{(0)}.$$

We assume that $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(i+1)}$ are the input and output variables of $\boldsymbol{F}^{(i)}$, where $\boldsymbol{x}^{(i)} = (x_0^{(i)}, \cdots, x_{t_i-1}^{(i)})$. Each $x_j^{(i)}$ is a variable over $\mathbb{F}_{2^n}$. For a given pair of $(\boldsymbol{u}^{(i)}, \boldsymbol{u}^{(i+1)})$, if the polynomial representation of $\boldsymbol{F}^{(i)}$ is available, one can determine whether $\pi_{\boldsymbol{u}^{(i)}}(\boldsymbol{x}^{(i)}) \to \pi_{\boldsymbol{u}^{(i+1)}}(\boldsymbol{x}^{(i+1)})$. We emphasize that $\pi_{\boldsymbol{u}^{(i)}}(\boldsymbol{x}^{(i)}) \to \pi_{\boldsymbol{u}^{(i+1)}}(\boldsymbol{x}^{(i+1)})$ if and only if the coefficient of $\pi_{\boldsymbol{u}^{(i)}}(\boldsymbol{x}^{(i)})$ in $\pi_{\boldsymbol{u}^{(i+1)}}(\boldsymbol{x}^{(i+1)})$ is a

constant $c \neq 0$. If there exists a trail such that

$$\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \to \cdots \pi_{\boldsymbol{u}^{(i)}}(\boldsymbol{x}^{(i)}) \to \cdots \to \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)}),$$

there exists a trail connecting $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)})$ and $\pi_{\boldsymbol{u}^{(r+1)}}(\boldsymbol{x}^{(r+1)})$, which naturally leads to the definition of *general monomial trail*.

Definition 2 (General Monomial Trail). *Let $\boldsymbol{F}^{(i)}$ be a sequence of polynomials over $\mathbb{F}_{2^n}$ for $0 \leq i < r$, while $\boldsymbol{x}^{(i+1)} = \boldsymbol{F}^{(i)}(\boldsymbol{x}^{(i)})$. We call a sequence of monomials $(\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}), \pi_{\boldsymbol{u}^{(1)}}(\boldsymbol{x}^{(1)}), \cdots, \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)}))$ an r-round general monomial trail connecting $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)})$ and $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ with respect to the composite function $\boldsymbol{F} = \boldsymbol{F}^{(r-1)} \circ \boldsymbol{F}^{(r-2)} \circ \cdots \circ \boldsymbol{F}^{(0)}$ if*

$$\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \to \pi_{\boldsymbol{u}^{(1)}}(\boldsymbol{x}^{(1)}) \to \cdots \to \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)}).$$

If there is at least one general monomial trail connecting $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)})$ and $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$, we write $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \rightsquigarrow \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$. Otherwise, $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \not\rightsquigarrow \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$. When $n = 1$, general monomial trail is equivalent to monomial trail.

Proposition 2. *$\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \rightsquigarrow \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ if $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \to \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$, and thus $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \not\rightsquigarrow \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ implies $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \not\to \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$.*

Proof. We proceed by induction on r. Assuming that this proposition holds for $r < s$, we now prove that it also holds for $r = s$. When $r = s$, we expand $\pi_{\boldsymbol{u}^{(s)}}(\boldsymbol{x}^{(s)})$ on $\pi_{\boldsymbol{u}^{(s-1)}}(\boldsymbol{x}^{(s-1)})$ as

$$\pi_{\boldsymbol{u}^{(s)}}(\boldsymbol{x}^{(s)}) = \bigoplus_{\pi_{\boldsymbol{u}^{(s-1)}}(\boldsymbol{x}^{(s-1)}) \to \pi_{\boldsymbol{u}^{(s)}}(\boldsymbol{x}^{(s)})} \varphi(\boldsymbol{u}^{(s-1)}) \cdot \pi_{\boldsymbol{u}^{(s-1)}}(\boldsymbol{x}^{(s-1)}), \varphi(\boldsymbol{u}^{(s-1)}) \neq 0.$$

Since $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \to \pi_{\boldsymbol{u}^{(s)}}(\boldsymbol{x}^{(s)})$, there is at least one $\pi_{\boldsymbol{u}^{(s-1)}}(\boldsymbol{x}^{(s-1)})$ contained by $\pi_{\boldsymbol{u}^{(s)}}(\boldsymbol{x}^{(s)})$ satisfying $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \to \pi_{\boldsymbol{u}^{(s-1)}}(\boldsymbol{x}^{(s-1)})$. According to the assumption that $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \rightsquigarrow \pi_{\boldsymbol{u}^{(s-1)}}(\boldsymbol{x}^{(s-1)})$, we have $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \rightsquigarrow \pi_{\boldsymbol{u}^{(s)}}(\boldsymbol{x}^{(s)})$. □

Example 1. Let $x_0, x_1, y, z \in \mathbb{F}_{2^3}$ with the irreducible polynomial $f(x) = x^3 + x + 1$. $z = 2y^3$, $y = x_0^3 \oplus 2x_0 \oplus x_1^2$.

Considering the monomial x_0^5, we can compute all the monomials of y as

$$\begin{aligned}
y^0 &\equiv 1,\\
y^1 &\equiv x_0^3 \oplus 2x_0 \oplus x_1^2,\\
y^2 &\equiv x_0^6 \oplus 4x_0^2 \oplus x_1^4,\\
y^3 &\equiv 2x_0^7 \oplus x_0^6x_1^2 \oplus \underline{4x_0^5} \oplus x_0^3x_1^4 \oplus 3x_0^3 \oplus 4x_0^2x_1^2 \oplus x_0^2 \oplus 2x_0x_1^4 \oplus x_1^6,\\
y^4 &\equiv \underline{x_0^5} \oplus 6x_0^4 \oplus x_1,\\
y^5 &\equiv 6x_0^7 \oplus 2x_0^6 \oplus x_0^5x_1^2 \oplus \underline{7x_0^5} \oplus 6x_0^4x_1^2 \oplus x_0^3x_1 \oplus 2x_0x_1 \oplus x_0 \oplus x_1^3,\\
y^6 &\equiv 4x_0^7 \oplus x_0^6x_1 \oplus 6x_0^6 \oplus x_0^5x_1^4 \oplus 6x_0^4x_1^4 \oplus x_0^4 \oplus 6x_0^3 \oplus 4x_0^2x_1 \oplus x_1^5
\end{aligned}$$

$$y^7 \equiv 6x_0^7x_1^4 \oplus 4x_0^7x_1^2 \oplus 2x_0^7x_1 \oplus 2x_0^6x_1^4 \oplus x_0^6x_1^3 \oplus 6x_0^6x_1^2 \oplus 6x_0^6 \oplus x_0^5x_1^6 \oplus 7x_0^5x_1^4,$$
$$\oplus 4x_0^5x_1 \oplus \underline{2x_0^5} \oplus 6x_0^4x_1^6 \oplus x_0^4x_1^2 \oplus 7x_0^4 \oplus x_0^3x_1^5 \oplus 6x_0^3x_1^2 \oplus 3x_0^3x_1 \oplus 4x_0^3$$
$$\oplus 4x_0^2x_1^3 \oplus x_0^2x_1 \oplus 6x_0^2 \oplus 2x_0x_1^5 \oplus x_0x_1^4 \oplus 3x_0 \oplus x_1^7.$$

Similarly, we can compute all the monomial of z

$$z^0 \equiv y^0,\ z^1 \equiv \underline{2y^3},\ z^2 \equiv 4y^6, z^3 \equiv 4y^3 \equiv 3y^2,$$

$$z^4 \equiv 6y^{12} \equiv \underline{6y^5}, z^5 \equiv 7y^{15} \equiv 7y, z^6 \equiv 5y^{18} \equiv \underline{5y^4}, z^7 \equiv y^{21} \equiv \underline{y^7}.$$

There are four monomial trails connecting x_0^5 and monomials of z:

$$x_0^5 \to y^3 \to z^1,\quad x_0^5 \to y^4 \to z^6,\quad x_0^5 \to y^5 \to z^4,\quad x_0^5 \to y^7 \to z^7.$$

Comparison with word-based division property. At a first glance, the general monomial prediction is similar to the word-based division property as both of them are described at the *word* level. However, we emphasize that they are completely different, especially in the way of extracting information. While the word-based division property can only exploit the information of the degree, our general monomial prediction can essentially utilize the internal structure of the ciphers in a more fine-grained way. Actually, it is more like the bit-based division property since *word* is the minimum unit of polynomials over $\mathbb{F}_{2^n}$.

Comparison with bit-based division property. From the example above we can see that the obvious difference between the general monomial prediction and bit-based division property is the range of the variables. Given a specific monomial m, there are two possible cases for the coefficient c in the ANF of a block cipher: $c = 1$ or $c = 0$, i.e., the ANF contains exactly m or the ANF does not contain m. However, since the coefficient c for ciphers over fields ranges over the 2^n elements of $\mathbb{F}_{2^n}$, the existence of a monomial m represents multiple states. As long as $c \neq 0$, the monomial m does appear in the polynomial representations. Recall that two-subset bit-based division property can essentially allow us to derive one of two possible results: the ANF of a block cipher does not contain any multiple of the monomial m, or we do not know any thing about the monomial. Given a specific monomial m for ciphers over fields, we can derive one of the following two results for the general monomial prediction according to Proposition 2:

- The monomial m with a corresponding coefficient $c \neq 0$ does not appear in the polynomial representation if there is no general monomial trail from m to the polynomial representation of the block cipher,
- We do not know anything about the monomial.

Essentially, we believe that the concept of the general monomial prediction is more common with the conventional bit-based division property. Moreover, we emphasize that due to the field-based structure for ciphers described over $\mathbb{F}_{2^n}$, the word-based/bit-based division property fails to be useful in this case.

3.2 Propagation Rules of Basic Field-Based Operations

Considering a sequence of monomials

$$(\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}), \pi_{\boldsymbol{u}^{(1)}}(\boldsymbol{x}^{(1)}), \cdots \pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})),$$

where $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(i+1)}$ are the input and output of $\boldsymbol{F}^{(i)}$. Each pair $(\boldsymbol{u}^{(i)}, \boldsymbol{u}^{(i+1)})$ is a valid monomial trail through $\boldsymbol{F}^{(i)}$ if and only if $\pi_{\boldsymbol{u}^{(i)}}(\boldsymbol{x}^{(i)}) \to \pi_{\boldsymbol{u}^{(i+1)}}(\boldsymbol{x}^{(i+1)})$ (Notice that $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(i+1)}$ are only symbolic variables.). However, each $\boldsymbol{u}^{(i)}$ is defined in $\mathbb{F}_{2^n}$ where the size of n is typically larger than 32. So we are not able to depict the possible propagations by simple observation or exhaustive search. As any arithmetization-oriented cipher can be represented as a sequence of basic operations such as XOR, AND, COPY, m-COPY, and POWER, we carefully investigate the arithmetical feature of operations and prove the propagation rules. Our propagation rules put no restrictions on the irreducible polynomial since we do not care about the exact value of the coefficients.

Rule 1 (Field-based XOR) *Let F be a function compressed by an XOR over $\mathbb{F}_{2^n}$, where the input $\boldsymbol{x} = (x_0, x_1, \cdots, x_{n-1})$ and the output $\boldsymbol{y}$ is calculated as $\boldsymbol{y} = (x_0 \oplus x_1, x_2, \cdots, x_{n-1})$. Considering a monomial of $\boldsymbol{x}$ as $\boldsymbol{x}^{\boldsymbol{u}}$, the monomial $\boldsymbol{y}^{\boldsymbol{v}}$ contains $\boldsymbol{x}^{\boldsymbol{u}}$ iff*

$$\boldsymbol{v} = (v, u_2, \cdots, u_{n-1}),$$

where $v = u_0 + u_1,\ v \succeq u_0$.

Proof. We have

$$(x_0 \oplus x_1)^v \equiv \bigoplus_{0 \le u_0 \le v} p_v(u_0) \cdot (x_0^{u_0} x_1^{v-u_0}),$$

where $p_v(u_0) = 1$ if $\binom{v}{u_0}$ is odd and $p_v(u_0) = 0$ if $\binom{v}{u_0}$ is even. $\binom{v}{u_0}$ is the binomial coefficient. Clearly, $\binom{v}{u_0}$ is odd if and only if $u_0 \preceq v$ according to the Lucas's theorem. Therefore, if $x_0^{u_0} \cdot x_1^{u_1} \to (x_0 \oplus x_1)^v$, there must be $p_v(u_0) = 1$ and we have $u_0 + u_1 = v, u_0 \preceq v$. Conversely, if $u_0 + u_1 = v, u_0 \preceq v$, we have $p_v(u_0) = 1$ and $x_0^{u_0} \cdot x_1^{u_1} \to (x_0 \oplus x_1)^v$. □

Rule 2 (Field-based AND) *Let F be a function compressed by an AND over $\mathbb{F}_{2^n}$, where the input $\boldsymbol{x} = (x_0, x_1, \cdots, x_{n-1})$ and the output $\boldsymbol{y}$ is calculated as $\boldsymbol{y} = (x_0 x_1, x_2, \cdots, x_{n-1})$. Considering a monomial of $\boldsymbol{x}$ as $\boldsymbol{x}^{\boldsymbol{u}}$, the monomial $\boldsymbol{y}^{\boldsymbol{v}}$ contains $\boldsymbol{x}^{\boldsymbol{u}}$ iff*

$$\boldsymbol{v} = (u_0, u_2, \cdots, u_{n-1}),$$

where $(u_0, u_1) = (i, i)$, for $0 \le i \le 2^n - 1$.

Proof. Since

$$(x_0 x_1)^v = (x_0 x_1)^{u_0} = x_0^{u_0} x_1^{u_0} = x_0^{u_0} x_1^{u_1},$$

we have $x_0^{u_0} x_1^{u_1} \to (x_0 x_1)^v$ if $v = u_0 = u_1 = i$ for $0 \le i \le 2^n - 1$. Conversely if $v = u_0 = u_1 = i$ for $0 \le i \le 2^n - 1$, there must be $x_0^{u_0} x_1^{u_1} = (x_0 x_1)^i = (x_0 x_1)^v$ and $x_0^{u_0} x_1^{u_1} \to (x_0 x_1)^v$. □

Rule 3 (Field-based COPY) *Let F be a COPY function over $\mathbb{F}_{2^n}$, where the input $\boldsymbol{x} = (x_0, x_1, \cdots, x_{n-1})$ and the output $\boldsymbol{y}$ is calculated as $\boldsymbol{y} = (x_0, x_0, x_1, \cdots, x_{n-1})$. Considering a monomial of $\boldsymbol{x}$ as $\boldsymbol{x}^{\boldsymbol{u}}$, the monomial $\boldsymbol{y}^{\boldsymbol{v}}$ contains $\boldsymbol{x}^{\boldsymbol{u}}$ iff*

$$\boldsymbol{v} = (v_0, v_1, u_1, u_2, \cdots, u_{n-1}),$$

where

$$(v_0, v_1) = \begin{cases} (0,0), & \text{if } u_0 = 0; \\ (i, u_0 - i), (j, u_0 + 2^n - 1 - j), & \text{else.} \end{cases}$$

for $0 \le i \le u_0, u_0 \le j \le 2^n - 1$.

Proof. For $u_0 = 0$, if $x_0^{u_0} \to x_0^{v_0+v_1}$, there must be $v_0 + v_1 = 0$, which implies $(v_0, v_1) = (0, 0)$. Conversely if $u_0 = 0$ and $(v_0, v_1) = (0, 0)$, we have $x_0^{u_0} = x_0^{v_0+v_1} = x_0^0$ and $x_0^{u_0} \to x_0^{v_0+v_1}$.
Let us now consider $u_0 \neq 0$. When $v_0 + v_1 \le 2^n - 1$, if $x_0^{u_0} \to x_0^{v_0+v_1}$ we have $u_0 = v_0 + v_1$ and it holds that $(v_0, v_1) = (i, u_0 - i)$ for $0 \le i \le u_0$. Conversely if $(v_0, v_1) = (i, u_0 - i)$ for $0 \le i \le u_0$, we have $v_0 + v_1 = u_0$ and $x_0^{u_0} \to x_0^{v_0+v_1}$.
When $v_0 + v_1 > 2^n - 1$, we have $v_0 + v_1 = t + 2^n - 1$ and $x_0^{v_0+v_1} \equiv x^t$, $0 < t \le 2^n - 1$. If $x_0^{u_0} \to x_0^{v_0+v_1}$, we have $x_0^{u_0} \to x_0^t$ and $u_0 = t$. Therefore it holds that $(v_0, v_1) = (j, u_0 + (2^n - 1) - j)$ for $u_0 \le j \le 2^n - 1$. Conversely if $(v_0, v_1) = (j, u_0 + (2^n - 1) - j)$ for $u_0 \le j \le 2^n - 1$, we have $v_0 + v_1 = u_0 + 2^n - 1$ and $x_0^{u_0} \equiv x_0^{u_0+2^n-1} \equiv x_0^{v_0+v_1}$. Then $x_0^{u_0} \to x_0^{v_0+v_1}$. □

Rule 4 (Field-based POWER) *Let F be a POWER function over $\mathbb{F}_{2^n}$, where the input $\boldsymbol{x} = (x_0, x_1, \cdots, x_{n-1})$ and the output $\boldsymbol{y}$ is calculated as $\boldsymbol{y} = (x_0^d, x_1, \cdots, x_{n-1})$, for $\gcd(d, 2^n - 1) = 1$. Considering a monomial of $\boldsymbol{x}$ as $\boldsymbol{x}^{\boldsymbol{u}}$, the monomial $\boldsymbol{y}^{\boldsymbol{v}}$ contains $\boldsymbol{x}^{\boldsymbol{u}}$ iff*

$$\boldsymbol{u} = (v, u_1, \cdots, u_{n-1}),$$

where

$$v = \begin{cases} u_0, & \text{if } u_0 = 0 \text{ or } 2^n - 1; \\ (d^{-1})u_0 \bmod (2^n - 1), & \text{else.} \end{cases}$$

Proof. For $u_0 = 0$, if $(x_0)^0 \to (x_0)^{dv}$ then v must be 0. Conversely if $v = 0$, we have $(x_0)^{u_0} \to (x_0)^{dv}$. For $u_0 = 2^n - 1$, if $(x_0)^{2^n-1} \to (x_0)^{dv}$, then v must be $2^n - 1$. Conversely if $v = 2^n - 1$, we have $(x_0)^{d\times(2^n-1)} \equiv x_0^{2^n-1}$ and $x_0^{2^n-1} \to x_0^{2^n-1}$.
If $u_0 \neq 0$ and $u_0 \neq 2^n - 1$, if $x_0^{u_0} \to (x_0^d)^v$ we have $u_0 = dv \mod (2^n - 1)$. Conversely if $u_0 = dv \mod (2^n - 1)$, we have $(x_0^d)^v \equiv x_0^{dv \mod (2^n-1)} \equiv x_0^{u_0}$. Therefore $(x_0)^{u_0} \to (x_0)^{dv}$. Then we have $v = (d^{-1})u_0 \mod (2^n - 1)$. □

Rule 5 (Field-based m-COPY) *Let F be a m-COPY function over $\mathbb{F}_{2^n}$, where the input $\boldsymbol{x} = (x_0, x_1, \cdots, x_{n-1})$ and the output $\boldsymbol{y}$ is calculated as $\boldsymbol{y} = (\underbrace{x_0, \cdots, x_0}_{m},$*

$x_1, \cdots, x_{n-1}$). *Considering a monomial of* $\boldsymbol{x}$ *as* $\boldsymbol{x^u}$*, the monomial* $\boldsymbol{y^v}$ *contains* $\boldsymbol{x^u}$ *iff*

$$\boldsymbol{v} = (v_0, v_1, \cdots, v_{m-1}, u_1, u_2, \cdots, u_{n-1}),$$

where

$$(v_0, \cdots, v_{m-1}) = \begin{cases} (0, 0, \cdots, 0), & \text{if } u_0 = 0; \\ (i_0^s, \cdots, i_{m-2}^s, i_{m-1}^s) \text{ for } 0 \le s \le m-1, & \text{else.} \end{cases}$$

Here, $i_{m-1}^s = u_0 + (s-1)(2^n - 1) - \sum_{j=0}^{m-2} i_j^s$, $0 \le i_j^s \le 2^n - 1$ *for* $0 \le j < m$.

Proof. For $u_0 = 0$, if $x_0^0 \to x_0^{v_0+v_1+\cdots+v_{m-1}}$ there must be $(v_0, v_1, \cdots, v_{m-1}) = (0, 0, \cdots, 0)$. Conversly, if $(v_0, v_1, \cdots, v_{m-1}) = (0, 0, \cdots, 0)$ we have $(x_0)^{u_0} \to (x_0)^{v_0+v_1+\cdots+v_{m-1}}$.
Let us consider $u_0 \neq 0$. We have $0 < v_0 + v_1 + \cdots + v_{m-1} \le m \cdot (2^n - 1)$. When $s(2^n - 1) < v_0 + v_1 + \cdots + v_{m-1} \le (s+1)(2^n - 1)$, $0 \le s \le m-1$, we have $v_0 + \cdots + v_{m-1} = t + (s-1)(2^n - 1)$, $0 < t \le 2^n - 1$.
If $x_0^{u_0} \to x_0^{v_0+v_1+\cdots+v_{m-1}}$, we have $x_0^{u_0} \to x_0^t$ and thus $u_0 = t$. Therefor it holds that $(v_0, v_1, \cdots, v_{m-1}) = (i_0^s, i_1^s, \cdots, i_{m-2}^s, u_0 + (s-1)(2^n - 1) - \sum_{j=0}^{m-2} i_j^s)$ for $0 \le i_j^s \le 2^n - 1$.
Conversely if $(v_0, v_1, \cdots, v_{m-1}) = (i_0^s, i_1^s, \cdots, i_{m-2}^s, u_0 + (s-1)(2^n - 1) - \sum_{j=0}^{m-2} i_j^s)$ for $0 \le i_j^s \le 2^n - 1$, we have $v_0 + v_1 + \cdots + v_{m-1} = u_0 + (s-1)(2^n - 1)$ and $x_0^{u_0} \equiv x_0^{u_0+(s-1)(2^n-1)} \equiv x_0^{v_0+v_1+\cdots+v_{m-1}}$. Then $x^{u_0} \to x_0^{v_0+v_1+\cdots+v_{m-1}}$. □

Example 2. Let $x_0, x_1, y, z \in \mathbb{F}_{2^3}$ with the irreducible polynomial $f(x) = x^3 + x + 1$. $y = (x_0 \oplus 3x_1)^3$. Compute (u_0, u_1) when $x_0^{u_0} \cdot x_1^{u_1} \rightsquigarrow y^v, v = 2$.

Consider $z = x_0 \oplus 3x_1$, then $y = z^3$. Then we need to compute all the monomial trails $x_0^{u_0} x_1^{u_1} \rightsquigarrow z^w \rightsquigarrow y^v$. According the Rule 4, we have $w = 3v \mod (7) = 6 \mod (7)$, $w = 6$. As $w = u_0 + u_1$ and $u_0 \preceq w$, we have $u_0 = 0, 2, 4, 6$. Then we deduce that $(u_0, u_1, w, v) = (6, 0, 6, 2), (4, 2, 6, 2), (2, 4, 6, 2), (0, 6, 6, 2)$ by the propagation rules. It is verified by

$$y^2 = x_0^6 \oplus 5x_0^4 \cdot x_1^2 \oplus 7x_0^2 \cdot x_1^4 \oplus 6x_1^6 \to (u_0, u_1, v) = (6, 0, 2), (4, 2, 2), (2, 4, 2), (0, 6, 2).$$

3.3 Bit-Vector Models for Field-Based Operations

In this subsection, we take advantage of the bit-theory of SMT and translate the propagations into a system of equations involving both arithmetic operations and bit-based operations. The solutions to the constraints are all the possible monomial trails through the basic operations. Moreover, we avoid arithmetic multiplications and arithmetic modular to obtain efficient bit-vector constraints. The models for XOR, AND, COPY, m-COPY, and POWER are introduced as follows.

Model 1 (Field-based XOR) *Let* $(u_0, u_1) \xrightarrow{XOR} (v)$ *denote the monomial trails through the field-based XOR function, where two n-bit words are compressed to one n-bit word using an XOR operation. Then, it can be depicted using the following constraints:*

$$\begin{cases} u_0 + u_1 = v, \\ v \ \wedge \ u_0 = u_0, \\ u_0, u_1, v \text{ are } n\text{-bit variables.} \end{cases}$$

The constraint $v \ \wedge \ u_0 = u_0$ excludes the invalid trails for $v \not\succeq u_0$.

Model 2 (Field-based AND) *Let* $(u_0, u_1) \xrightarrow{AND} (v)$ *denote the monomial trails through the field-based AND function, where two n-bit words are compressed to one n-bit word using an AND operation. Then, it can be depicted using the following constraints:*

$$\begin{cases} u_0 = v, \\ u_1 = v, \\ u_0, u_1, v \text{ are } n\text{-bit variables.} \end{cases}$$

Model 3 (Field-based COPY) *Let* $(u) \xrightarrow{COPY} (v_0, v_1)$ *denote the monomial trails through the field-based COPY function, where one n-bit word is copied to two n-bit words using a COPY operation. Then, it can be depicted using the following constraints:*

$$\begin{cases} v_0 + v_1 + t = t \ || \ u, \\ u \ || \ t \neq 0^n \ || \ 1^1, \\ u, v_0, v_1 \text{ are } n\text{-bit variables,} \\ t \textit{ is a } 1\text{-bit variable.} \end{cases}$$

Proof. For $u = 0$, the only valid trail is $v_0 + v_1 = 0$ since $t \neq 1$. For $u \neq 0$, we have $v_0 + v_1 = u$ when $t = 0$ and $v_0 + v_1 + 1 = 1 \ || \ u = u + 2^n$ when $t = 1$. □

Model 4 (Field-based m-COPY) *Let* $(u) \xrightarrow{m-COPY} (v_0, v_1, \cdots, v_{m-1})$ *denote the monomial trails through the field-based m-COPY function, where one n-bit word is copied to m n-bit words using an m-COPY operation. Then, it can be depicted using the following constraints:*

$$\begin{cases} v_0 + v_1 \cdots + v_{m-1} + t = t \ || \ u, \\ u \ || \ t \neq 0^n \ || \ q, \ 0 < q \leq m - 1 \\ t \leq m - 1, \\ u, v_0, v_1 \text{ are } n\text{-bit variables,} \\ t \text{ is a } s\text{-bit variable}, s = \lfloor \log_2(m-1) \rfloor + 1. \end{cases}$$

The constraints

$$u \mid\mid t \neq 0^n \mid\mid q,\ 0 < q \leq m-1$$

is implemented in STP solver with an IF-THEN-ELSE branch statement as follows

$$\text{ASSERT (IF } u = 0^n \text{ THEN } t = 0^s \text{ ELSE } t \geq 0^s);$$

Model 5 (Field-based POWER) *Let* $(u) \xrightarrow{POWER} (v)$ *denote the monomial trails through the field-based POWER function, where one* n*-bit word is transmitted to another* n*-bit word using a POWER operation,* $\gcd(d, 2^n - 1) = 1$*, Then, it can be depicted using the following constraints:*

$$\begin{cases} d \times v + t = t \mid\mid u, \\ t \leq d-1, \\ u, v \text{ are } n\text{-bit variables}, \\ t \text{ is an } s\text{-bit variable}, s = \lfloor log_2(d-1) \rfloor + 1. \end{cases}$$

Moreover, when $d = 2^l + 1$ *or* $d = 2^l - 1$*, we can avoid multiplications and give more efficient constraints as:*

$$\begin{cases} (v \ll l) \pm v + t = t \mid\mid u, \\ t \leq d-1, \\ u, v \text{ are } n\text{-bit variables}, \\ t \text{ is an } s\text{-bit variable}, s = \lfloor log_2(d-1) \rfloor + 1. \end{cases}$$

Proof. When $u = 0$, we have $d \times v = t \times (2^n - 1)$. Since $\gcd(d, 2^n - 1) = 1$, we have $\gcd(d, t \times (2^n - 1)) \leq t < d$, then d is not divisible by $t \times (2^n - 1)$ if $t \neq 0$. Then we have $v = 0$.

When $u = 2^n - 1$, $d \times v = (1 + t) \times (2^n - 1)$. If $t = d - 1$, $v = 2^n - 1$. If $t \neq d - 1$, there are no solutions since d is not divisible by $(t + 1) \times (2^n - 1)$ for $0 < t + 1 < d$.

When $u \neq 0$ and $u \neq 2^n - 1$, we have $d \times v = u + t \times (2^n - 1)$ and thus $u = dv \mod (2^n - 1)$. □

3.4 Detecting the Upper Bound of the Algebraic Degree

In this subsection, we describe how to detect the upper bound of the algebraic degree for block ciphers considering round keys. All the round keys $\boldsymbol{k}^{(i)}$ are regarded as independent input variables defined over $\mathbb{F}_{2^n}$ for $0 \leq i < r$. Suppose the input of the statement is defined over $\mathbb{F}_{2^n}^t$, that is, the length of the word size is n and the number of words is t. For $0 \leq i < r$, let $\pi_{\boldsymbol{u}^{(i)}}(\boldsymbol{x}^{(i)})$ denote the input monomials of the i-th round function, respectively. Then $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ denotes a monomial of ciphertext we are interested in and is usually set as a unit vector to study a certain word of the ciphertext in practice. $\pi_{\boldsymbol{v}^{(i)}}(\boldsymbol{k}^{(i)})$ denotes the monomial of the i-th round key. We use equations to add constraints for variables

$\boldsymbol{u}^{(i)}$, $\boldsymbol{u}^{(i+1)}$, and $\boldsymbol{v}^{(i)}$ according to the function between them. The monomial trails through the public function described in the models are introduced in Section 3.3. Notice that the monomials $\pi_{\boldsymbol{v}^{(i)}}(\boldsymbol{k}^{(i)})$ are treated equivalently as $\pi_{\boldsymbol{u}^{(i)}}(\boldsymbol{x}^{(i)})$ when we add constraints.

Initial constraints. According to Proposition 2, if we want to determine whether a specific monomial $\pi_{\tilde{\boldsymbol{u}}^{(r)}}(\boldsymbol{x}^{(r)})$ does not contain any (key-related) monomial $\pi_{\boldsymbol{v}^{(0)},\cdots,\boldsymbol{v}^{(r)}}(\boldsymbol{k}^{(0)},\cdots,\boldsymbol{k}^{(r)}) \cdot \pi_{\tilde{\boldsymbol{u}}^{(0)}}(\boldsymbol{x}^{(0)})$, we only need to check whether there exist some trails from the monomial $\pi_{\boldsymbol{v}^{(0)},\cdots,\boldsymbol{v}^{(r)}}(\boldsymbol{k}^{(0)},\cdots,\boldsymbol{k}^{(r)}) \cdot \pi_{\tilde{\boldsymbol{u}}^{(0)}}(\boldsymbol{x}^{(0)})$ to $\pi_{\tilde{\boldsymbol{u}}^{(r)}}(\boldsymbol{x}^{(r)})$. Given an initial vector $I_u = (\tilde{u}_0^{(0)},\cdots,\tilde{u}_{t-1}^{(0)})$, where $\tilde{u}_i^{(0)} \in \mathbb{F}_{2^n}$, we use

$$u_i^{(0)} = \tilde{u_i}^{(0)} \text{ for } 0 \le i < t$$

to add the initial constraints on $\boldsymbol{u}^{(0)}$ and search for the general monomial trails. Notice that we do not add any constraints on $(\boldsymbol{v}^{(0)},\cdots,\boldsymbol{v}^{(r)})$ since they are all free variables over $\mathbb{F}_{2^n}$.

However, in the higher-order differential attacks [29], we are interested in the algebraic degree of F. If the algebraic degree of F is $\delta(F)$, then we have $\bigoplus_{v\in\mathcal{V}\oplus c} F(v) = 0$ if the dimension of the affine vector space $\mathcal{V}\oplus c$ is strictly greater than $\delta(F)$. We then use Corollary 1 that the algebraic degree of monomial $x_0^{u_0}\cdot\ldots\cdot x_{t-1}^{u_{t-1}}$ is given by $\sum_{i=0}^{t-1} wt(u_i)$. Therefore, if we want to determine the algebraic degree of a certain monomial $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$, we only need to check whether $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ contains any term in the set $\mathbb{S}_l$ for $d \le l \le \Delta$, where

$$\mathbb{S}_l = \{\pi_{\boldsymbol{v}^{(0)},\cdots,\boldsymbol{v}^{(r)}}(\boldsymbol{k}^{(0)},\cdots,\boldsymbol{k}^{(r)}) \cdot \pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)}) \mid \sum_{i=0}^{t-1} wt(u_i^{(0)}) = l\} \tag{4}$$

and Δ denotes the maximum algebraic degree. According to Proposition 2, if the monomial $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ contains no monomials in $\mathbb{S}_l$ for $d \le l \le \Delta$, the algebraic degree $\delta(F)$ is strictly less than d and the upper bound of the algebraic degree is $d-1$.

Stopping rules. If we consider the algebraic degree of the i'th ciphertext word, then we use

$$\begin{cases} u_i^{(r)} = 1, & \text{if } i = i', \\ u_i^{(r)} = 0, & \text{if } i \ne i'. \end{cases}$$

to add the stopping rules on $\boldsymbol{u}^{(r)}$.

Detecting the upper bound of the algebraic degree. Let us denote the stopping constraints as $\Gamma = (\underbrace{0,\cdots,0}_{i'},1,\underbrace{0,\cdots,0}_{t-i'-1})$. If we want to determine whether the upper bound of the algebraic degree for a certain monomial

Algorithm 1: $\delta = \mathsf{SearchDegree}(\mathcal{M}_r, \Delta, \Gamma)$

Input: The r-round SMT model $\mathcal{M}_r$, the maximum algebraic degree Δ, the stopping constraints Γ
Output: The algebraic degree δ

```
M ← M_r;
δ = 0;
for i = 0; i < t; i ← i + 1 do
    M.con ← u_i^(r) = Γ[i];
for i = Δ; i ≥ 0; i ← i − 1 do
    M.con ← Σ_j wt(u_j^(0)) = i;
    solve the r-round SMT model M;
    if the problem is satisfiable then
        δ = i;
        break;
return δ;
```

$\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ is $d-1$, we only need to check whether $\pi_{\boldsymbol{u}^{(r)}}(\boldsymbol{x}^{(r)})$ contains any term in the set $\mathbb{S}_l$. $\mathbb{S}_l$ is defined in Equation (4). For $l \leq \Delta$, if there is no general monomial trail from any monomial contained by $\mathbb{S}_l$ to $\pi_{\boldsymbol{u}}^{(r)}(\boldsymbol{x}^{(r)})$ for $d \leq l \leq \Delta$, the upper bound of the algebraic degree is $d-1$. Therefore, we use constraint

$$\sum_i wt(u_i^{(0)}) = l \tag{5}$$

from $l = \Delta$ in a decreasing order to add the initial constraint on $\boldsymbol{u}^{(0)}$. Δ denotes the theoretical upper bound of the algebraic degree with a maximum value of $n \cdot t - 1$ for permutations.

The framework of the whole algorithm is illustrated in Algorithm 1. When the SMT solver finds the solution for the first satisfiable problem, an assignment of the variables that makes the problem satisfiable is obtained. Then the searching process finishes and we have found the upper bound of the algebraic degree δ.

4 Theoretical Upper Bound for MiMC-like Constructions in (Unbalanced) Feistel Network

In this section, for a better insight into the behavior of the degree growth, we firstly investigate the algebraic degree of MiMC-like constructions in the (unbalanced) Feistel network. Based on the upper bound given in [22] valid for the MiMC-like construction in Even-Mansour schemes, we propose a new linear upper bound in Section 4.1. Besides, we show that higher-order differential distinguisher can be established using the special structure of the function in Section 4.2.

4.1 Theoretical Upper Bound on the Algebraic Degree

Considering $E_k\colon \mathbb{F}_{2^n}^t \to \mathbb{F}_{2^n}^t$ as an (unbalanced) Feistel-network cipher, $t \geq 2$. The j-th (expanding) round function is defined as

$$(x_0^{(j)}, x_1^{(j)}, \cdots, x_{t-1}^{(j)}) \leftarrow (x_1^{(j-1)} \oplus F(x_0^{(j-1)}), \cdots, x_{t-1}^{(j-1)} \oplus F(x_0^{(j-1)}), x_0^{(j-1)}), \quad (6)$$

where $F(x) := (x \oplus k^{(j-1)})^d$. $k = (k^{(0)}, \cdots, k^{(r-1)})$ denotes a sequence of round keys.

We firstly focus on the univariate degree of E_k. The maximum Hamming weight of the exponent for a single variable x_i in $x_j^{(r)}$ is represented by $\delta_{x_i}(x_j^{(r)})$.

Example 3. Let $y = x_0^2x_1^7 \oplus x_0^5x_1^3 \oplus x_0^8x_1^5$, then we have

$$\delta_{x_0}(y) = 2, \delta_{x_1}(y) = 3.$$

Recalling the upper bound given in [22] valid for the MiMC-like construction in Even-Mansour schemes, we apply this idea to the (unbalanced) Feistel network and prove the following Lemma 1.

Lemma 1. *Considering $E_k\colon \mathbb{F}_{2^n}^t \to \mathbb{F}_{2^n}^t$ as an (unbalanced) Feistel-network cipher represented as in Equation (6). For $0 \leq i, j < t$, $r \geq 1$, we have*

$$\delta_{x_i}(x_j^{(r)}) \leq \begin{cases} \min\{\lfloor \log_2(d^{r-(i+\theta(j))} + 1)\rfloor, n\}, & \text{if } r > i + \theta(j), \\ 1, & \text{else if } r = i - j + t \cdot \theta(j), \\ 0, & \text{else.} \end{cases}$$

where

$$\theta(j) = \begin{cases} 0, \text{ if } 0 \leq j < t-1, \\ 1, \text{ if } j = t-1. \end{cases}$$

Proof. Notice that the degree of x_i grows differently in the different branches of the (unbalanced) Feistel network. When $j < t-1$, we have that the maximum exponents of x_i in $x_j^{(i)}$ is d. Then the exponents of x_i in r-th round are upper bounded by d^{r-i} if $r > i$. This means that the upper bound of $\delta_{x_i}(x_j^{(r)})$ is the maximum integer l that satisfies $2^l - 1 \leq d^{r-i}$.Since $\delta_{x_i}(x_j^{(r)}) \leq n$ we have $\delta_{x_i}(x_j^{(r)}) \leq \min\{\lfloor \log_2(d^{r-i}+1)\rfloor, n\}$. For the case of $r \leq i$, by simple observation the maximum exponents of x_i in $x_j^{(i-j)}$ is 1 when $i > j$ and $x_j^{(r)}$ does not contain the variable x_i otherwise. Hence we have $\delta_{x_i}(x_j^{(r)}) = 1$ if $r = i - j$ and $\delta_{x_i}(x_j^{(r)}) = 0$ otherwise.

When $j = t-1$, due to the structure of the (unbalanced) Feistel network we have $\delta_{x_i}(x_{t-1}^{(r)}) = \delta_{x_i}(x_0^{(r-1)})$. Then we can derive $\delta_{x_i}(x_j^{(r)})$ in the same way before. Since the maximum exponents of x_i in $x_j^{(i+1)}$ is d, we have that the exponents in r-th round are upper bounded by $d^{r-(i+1)}$ and $\delta_{x_i}(x_j^{(r)}) \leq \min\{\lfloor \log_2(d^{r-(i+1)} + 1)\rfloor, n\}$ for $r > i+1$. For the case of $r \leq i$, the maximum exponents of x_i in $x_j^{(i-j+1)}$ is 1 and we have $\delta_{x_i}(x_j^{(r)}) = 1$ if $r = i - j + 1$.

□

Using Corollary 1, we propose the upper bound on the algebraic degree of MiMC-like constructions in (unbalanced) Feistel network as follows.

Proposition 3. *Considering* $E_k\colon \mathbb{F}_{2^n}^t \to \mathbb{F}_{2^n}^t$ *as a (unbalanced) Feistel-network cipher represented as in Equation* (6)*. Let* $\mathcal{R}_j = \log_d(2^n-1) + (t-1+\theta(j))$*, The algebraic degree of* E_k *satisfies that*

$$\delta(x_j^{(r)}) \leq \begin{cases} \sum_{i=0}^{t-1} \delta_{x_i}(x_j^{(r)}), & \text{if } r < \mathcal{R}_j, \\ t\cdot n - 1, & \text{else.} \end{cases}$$

The proof can be found in the full version of the paper [1].

Discussion of Proposition 3. By the proof of Proposition 3, we can see that the growth of the algebraic degree is almost linear. When $r \geq \mathcal{R}_j$, $\delta(x_i^{(r)})$ is always $t\cdot n-1$. However, we point out that it is not always the case. Taking the polynomial $(x_0 \oplus k_0)^{27}(x_1 \oplus k_1)^{18}$ as a simple example, the algebraic degree is $wt(27) + wt(18) = 6$ for the appearance of monomial $x_0^{27}x_1^{18}$. However, when $n = 3$, actually we have

$$(x_0 \oplus k_0)^{27}(x_1 \oplus k_1)^{18} \equiv (x_0 \oplus k_0)^{6}(x_1 \oplus k_1)^{4}$$

and the algebraic degree is $wt(6) + wt(4) = 4$. The deviation is caused by the offset of exponents when the degree is over $2^n - 1$, which is difficult to give a condition to guarantee when a particular monomial will change. Moreover, the relatively simple algebraic structure also leads to sparser terms and this decreases the practical degree.

4.2 Constructing Higher-Order Differential Distinguishers by Considering Different Numbers of Branches.

As is well known, if the algebraic degree of F is strictly smaller than $d-1$, then for any subspace $\mathcal{V} \subseteq \mathbb{F}_2^N$ with dimension $s \geq d$, we have $\bigoplus_{x\in\mathcal{V}\oplus c} F(x) = 0$. Unfortunately, the opposite does not hold in general. Even if the summing over all the $x \in \mathcal{V} \oplus c$ of dimension $s \leq d$ always results in a zero-sum, we cannot make sure if the algebraic degree is d since it can be caused by some special structure of the function. However, this provides us with a new approach for detecting the higher-order differential distinguisher. For multivariate polynomial $F_{\boldsymbol{k}}\colon \mathbb{F}_{2^n}^t \to \mathbb{F}_{2^n}$ defined as

$$F_{\boldsymbol{k}}(x_0, \cdots, x_{t-1}) = \sum_{\boldsymbol{u}=(u_0,\cdots,u_{t-1})\in\{0,1,\cdots,2^n-1\}^t} \varphi_{\boldsymbol{k}}(\boldsymbol{u}) \cdot \prod_{i=0}^{t-1} x_i^{u_i}, \boldsymbol{k} \in \mathbb{F}_{2^n}^r.$$

We limit ourselves to consider the sum of the Hamming weight for the exponents of different branches, denoted by $\sum_{i\in\mathbb{X}} \delta_{x_i}(x_j^{(r)})$ for some certain j. $\mathbb{X}$ represents the set of the branches we are interested in. Then by tracing the upper bound of $\sum_{i\in\mathbb{X}} \delta_{x_i}(x_j^{(r)})$ precisely, we can establish higher-order differential distinguishers. The following theorem is a corollary of Proposition 1 in [8].

Corollary 2. *Let* $F_{\boldsymbol{k}} : \mathbb{F}_{2^n}^t \to \mathbb{F}_{2^n}$ *be multivariate polynomial defined as*

$$F_{\boldsymbol{k}}(x_0, \cdots, x_{t-1}) = \sum_{\boldsymbol{u}=(u_0,\cdots,u_{t-1})\in\{0,1,\cdots,2^n-1\}^t} \varphi_{\boldsymbol{k}}(\boldsymbol{u}) \cdot \prod_{i=0}^{t-1} x_i^{u_i}, \boldsymbol{k} \in \mathbb{F}_{2^n}^r.$$

If there exist m *variables* $x_{j_0}, x_{j_1}, \cdots, x_{j_{m-1}}$ *satisfies that for each non-vanishing monomial in* $F_{\boldsymbol{k}}$ *there is* $\bigoplus_{w=0}^{m-1} hw(u_{j_w}) \leq s-1$*, we have* $\bigoplus_{v\in\mathcal{V}\oplus c} F_{\boldsymbol{k}}(v) = 0$*.* $\mathcal{V} = \{(l_0, l_1, \cdots, l_{t-1}) \mid (l_{j_0}, l_{j_1}, \cdots, l_{j_{m-1}}) \in V\}$ *for any affine subspace* $V \subseteq \mathbb{F}_{2^{m\times n}}$ *of dimension at least* s*.*

Proof. Each non-vanishing monomial of $F_{\boldsymbol{k}}$ can be written in the form of $\varphi_{\boldsymbol{k}}(\boldsymbol{u}) \cdot x_0^{u_0} x_1^{u_1} \cdot \cdots \cdot x_{t-1}^{u_{t-1}}$ with $\varphi_{\boldsymbol{k}}(\boldsymbol{u}) \neq 0$. Since the dimension of V is at least s, then we have

$$\sum_{(x_{j_0},\cdots,x_{j_{m-1}})\in V} x_{j_0}^{u_{j_0}} \cdot \ldots \cdot x_{j_{m-1}}^{u_{j_{m-1}}} = 0$$

and for each monomial of $F_{\boldsymbol{k}}$ we have

$$\sum_{(x_0,\cdots,x_{t-1})\in\mathcal{V}\oplus c} \varphi_{\boldsymbol{k}}(\boldsymbol{u}) \cdot \prod_{i=0}^{t-1} x_i^{u_i} = 0.$$

Consequently, $\bigoplus_{v\in\mathcal{V}\oplus c} F_{\boldsymbol{k}}(v) = 0$.

5 Applications to Feistel MiMC, MiMC and GMiMC

We apply our algorithm to some competitive arithmetization-oriented block ciphers, including MiMC and its generalization Feistel MiMC and GMiMC. All of them use $x \mapsto x^3$ as their round function, but are based on different design strategies. The original MiMC introduced by Albrecht et al. [3] is a family of block ciphers dedicated to applications that support operations in large finite fields posing largest performance bottleneck. Due to its outstanding performance in applications such as MPC, SNARKs and STARKs, it quickly became the optimal choice for many use cases. In the same specification, a variant of MiMC was proposed by inserting the original design into the Feistel structure, named Feistel MiMC or MiMC-$2n/n$. This first application of Feistel networks in `AO` ciphers brings more flexibility of being able to rely on a larger field size. In that spirit, Albrecht et al. proposed GMiMC [2], a family of block ciphers based on different types of Feistel networks which can operate on different numbers of branches.

5.1 Application to Feistel MiMC

In this subsection, we focus on Feistel MiMC, an r-round block cipher in Feistel network with n-bit block size and the same key size operating on $\mathbb{F}_{2^n}$. The i-th round function $F^{(i)}$ is depicted in Figure 1 and defined as

$$(x_0^{(i)}, x_1^{(i)}) \leftarrow (x_1^{(i-1)} \oplus (x_0^{(i-1)} \oplus k^{(i-1)})^d, x_0^{(i-1)})$$

$k = (k^{(0)}, \ldots, k^{(r-1)})$ denotes a sequence of r round subkeys. The round constants are omitted for simplicity since they can be regarded as part of the round keys and do not affect the upper bound of the algebraic degree. We denote Feistel MiMC specified by exponent d and block size n as $\mathsf{FeistelMiMC}_d(n, r)$. When $d = 3$, the number of rounds to achieve n-bit security is $r_n = 2n \cdot \log_3(2) + 1$ and the number of rounds to achieve $2n$-bit security is $r_N = \lceil 2n \cdot \log_3(2) \rceil + 3$. As far as we know, the best higher-order differential distinguisher in the literature is the 83-round one proposed in [8] for the permutation Feistel MiMC.

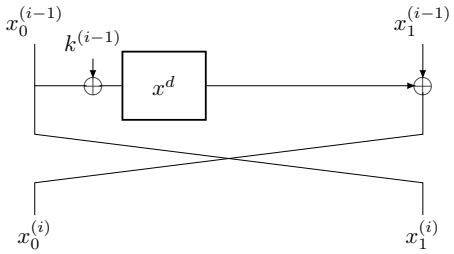

Fig. 1: The round function $F^{(i)}$ of $\mathsf{FeistelMiMC}_d(n, r)$.

Detect the algebraic degree of $\mathsf{FeistelMiMC}_d$ Let $\pi_{\boldsymbol{u}^{(0)}}(\boldsymbol{x}^{(0)})$ denote the monomials of the input statements of $\mathsf{FeistelMiMC}_d$. $\pi_{\boldsymbol{u}^{(i)}}(\boldsymbol{x}^{(i)})$ and $\pi_{\boldsymbol{v}^{(i)}}(\boldsymbol{k}^{(i)})$ denote the output statements of the i-th round function $F^{(i)}$ and the monomial of the i-th round key, respectively. We introduce auxiliary variables and the whole SMT model $\mathcal{M}_r$ is described as

$$\mathcal{M}_r \leftarrow \begin{cases} (u_0^{(i)}) \xrightarrow{COPY} (u_1^{(i+1)}, m^{(i)}) & for\ 0 \le i < r, \\ (v^{(i)}, m^{(i)}) \xrightarrow{XOR} (p_i) & for\ 0 \le i < r, \\ (p^{(i)}) \xrightarrow{POWER} (q^{(i)}) & for\ 0 \le i < r, \\ (q^{(i)}, u_1^{(i)}) \xrightarrow{XOR} (u_0^{(i+1)}) & for\ 0 \le i < r, \end{cases}$$

By setting the initial constraints and stopping rules, Algorithm 1 is implemented to search the algebraic degree of $\mathsf{FeistelMiMC}_d(n, r)$. We denote the algebraic degree of the left branch and the right branch by $\delta(x_0^{(r)})$ and $\delta(x_1^{(r)})$, respectively. Without loss of generality, we only search for $\delta(x_0^{(r)})$ due to the structure of the Feistel network, i.e., $\delta(x_1^{(r)}) = \delta(x_0^{(r-1)})$.

Comparison of our results with theoretical bounds. We have practically verified our results on small-scale instances of $\mathsf{FeistelMiMC}_3(n, r)$ with block size $n = 13$ and found that our detected bounds correspond to the practical results.

A concrete comparison of different degree bounds for $\delta(x_0)$ is given in Figure 2 for $\mathsf{FeistelMiMC}_3(129, r)$. The trivial upper bound is defined as 2^r. Meanwhile, we also depict the trivial lower bound to explicitly understand the security margin. Indeed, since the monomials $x_0^{3^r}$ and $x_1^{3^{r-1}}$ always appears in $x_0^{(r)}$ independently from the choice of round constants or round keys, we can define the trivial lower bound as $\max\{wt(3^r), wt(3^{r-1})\}$.

We notice that both our detected bound and our theoretical bound present a linear growth in the initial stage. A more substantial difference appears when the algebraic degree is reaching the maximal, namely when $r > \mathcal{R}_j$. While the theoretical bound predicts that $\delta(x_0^{(r)})$ reaches the maximal degree directly after the linear growth, the detected bounds show that there is still a long stage of slow growth before achieving the maximum algebraic degree. During some consecutive rounds, the algebraic degree even remains constant, called a *plateau* in [14]. Some plateaus cover a few rounds, e.g., $\delta(x_0^{(r)})$ stays constant at 254 for only 2 rounds. The largest plateau appears when the algebraic degree is $2n - 2$. It remains constant for an especially long time, covering a total of 27 rounds. Our results indicate that the stage of slow growth significantly influences the growth of the algebraic degree. Therefore, more rounds than previously predicted may be necessary to guarantee security against high-order differential distinguishers. For $\mathsf{FeistelMiMC}_3(129, r)$, our detected bound can produce the distinguisher for 124 rounds, which extends the theoretical distinguisher for 41 rounds.

We would like to mention that our Algorithm 1 can also be applied to search for the univariate degree $\delta_{x_i}(x_j^{(r)})$ by slightly modifying the initial constraints as

$$wt(u_i^{(0)}) = l$$

for $i \in \{0, 1\}$, respectively. Then if $wt(u_i^{(0)}) \leq s - 1$, we can always construct a higher-order differential distinguisher with a data complexity of 2^s for branch i. As an example, for $\mathsf{FeistelMiMC}_3(129, r)$, we can exhibit a distinguisher with data complexity $2^{127}/2^{129}$ for 82/83 rounds, both resulting in a zero-sum in the output of the right branch.

Comparison of our bounds with different exponents. We also applied our algorithm to different exponents d and observe the influence of exponents on the degree growth. A simple observation of Figure 3 is that the linear rate of the initial linear growth goes up with d and the number of rounds for the slow growth (i.e., from the round $\mathcal{R}_0$ until reaching the maximal algebraic degree) is reduced. The number of rounds for the slow growth is 42 rounds for $d = 3$ whereas it is 30 rounds for $d = 5$ despite the Hamming weight of d is the same. For $d = 31$, the number of rounds for the slow growth is only 15, with the largest plateau covering 10 rounds.

Known-key zero-sum distinguisher for the full $\mathsf{FeistelMiMC}_3(129, r)$. The known-key distinguishers for block ciphers were introduced by Knudsen and

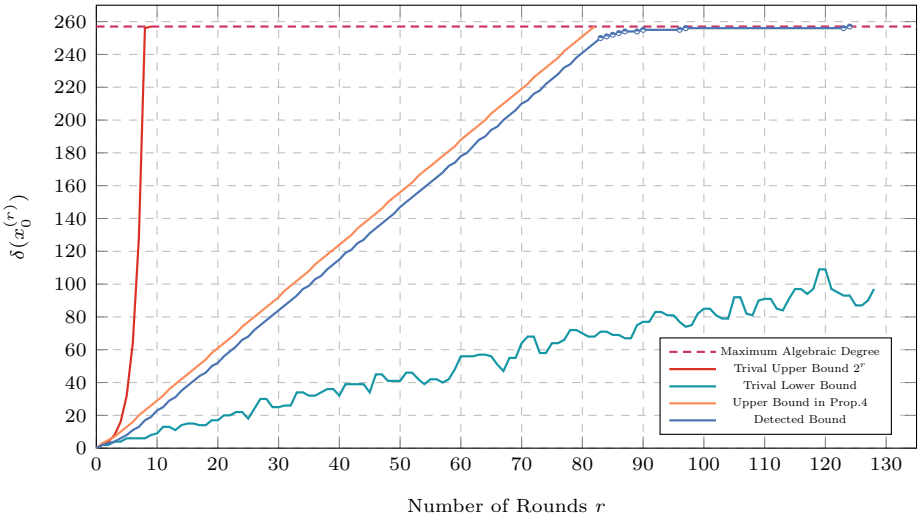

Fig. 2: Comparison of different degree bounds $\delta(x_0^{(r)})$ for $\mathsf{FeistelMiMC}_3(129, r)$.

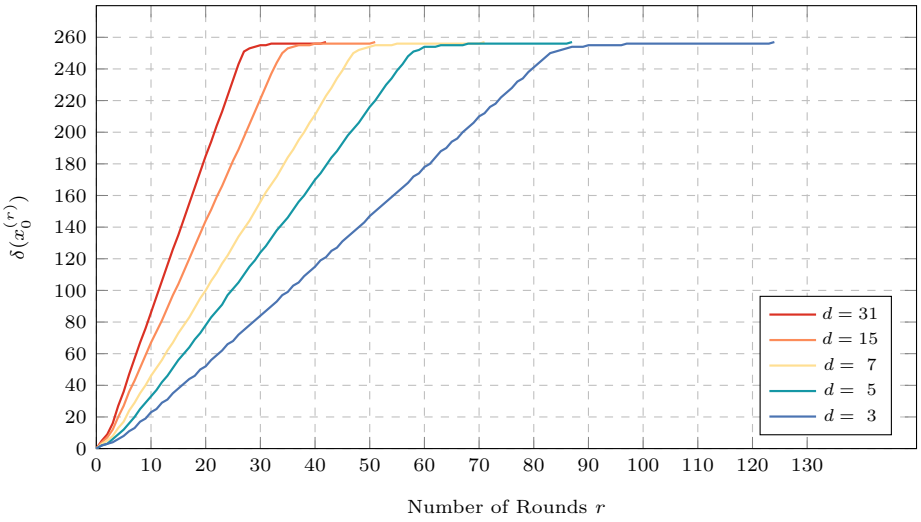

Fig. 3: Comparison of our degree bounds $\delta(x_0^{(r)})$ for $\mathsf{FeistelMiMC}_d(129, r)$ with different exponents d.

Rijmen at ASIACRYPT 2007 [28] and have been a major research direction in cryptanalysis since then. There is no secret material involved in the computation and the attacker aims to find a structural property for the cipher which an ideal cipher would not have. It is also related to the distinguishers for permutation since the analysis is often done in the known-key model. A well-known powerful

distinguisher for the known-key setting is the so-called zero-sum distinguisher [5]. The idea is based on the inside-out approach, where the attacker starts from the middle rounds and chooses a set of internal states so that the sum of the inputs and outputs are all zero when computing backwards and forwards.

Let us consider $\mathsf{FeistelMiMC}_3(129, r)$. By choosing a subspace of the right input branch of dimension 127, the distinguisher can be extended forwards for 82 rounds, with the output of the right branch achieving zero-sum property. The inverse of $\mathsf{FeistelMiMC}_3(129, r)$ still follows the Feistel network. When computing backwards, the active branch is now the left one. Then the distinguisher can be extended backwards for 81 rounds, resulting in a zero-sum property in the right output branch. This eventually leads to a distinguisher with complexity 2^{127} for a total of $82 + 81 = 163$ rounds. Besides, by saturating the branch x_1, we can further derive a zero-sum distinguisher of $83 + 82 = 165$ rounds. Moreover, since the upper bound of the algebraic degree $\delta(x_0^{(r)})$ is 250 for 83 rounds, we can establish a zero-sum distinguisher covering a total of $83 \times 2 = 166$ rounds by choosing a subspace of dimension $D = 251$. With the largest non-trivial vector space $\mathbb{F}_2^{257}$, we can deduce the longest zero-sum distinguisher covering a total of $124 \times 2 = 248$ rounds, much more than $r_N = 166$ rounds for $2n$-bit security.

5.2 Application to MiMC

In this subsection, we consider the algebraic degree of different variants of MiMC and investigate some possible choices for the generic exponents d. MiMC [3] is an r-round key-alternating block cipher with an n-bit block size and the same key size. Each round consists of three steps: a key addition with the master key k, a round constant addition of $c_i \in \mathbb{F}_{2^n}$, and the application of the non-linear function $R_d := x^d$ over $\mathbb{F}_{2^n}$ with $(d, 2^n - 1) = 1$. After r round iterations, an additional k is added at last. To simplify the representation, we equivalently regard $k \oplus c_i$ as the round key k_i and the instance $\mathsf{MiMC}_d(n, r)$ is defined by

$$\mathsf{MiMC}_d(n, r) := R_d(\cdots R_d(R_d(x \oplus k_0) \oplus k_1) \cdots) \oplus k_r \tag{7}$$

where $R_d(x) := x^d$.

Comparison of different choices for exponents d. Referring to the analysis proposed in MiMC, the best choice of the exponents seems to be of the form $d = 2^l - 1$ for integer l. We then apply our searching algorithm for $\mathsf{MiMC}_d(129, r)$ with $d \in \{3, 7, 15, 31\}$, respectively. Appendix A in [1] compares the different upper bounds of the algebraic degree. $\delta_{\mathrm{MP}}^{(d,r)}$ denotes the algebraic degree found by our algorithm, while $\delta_{[\mathrm{EGL}^+20]}^{(d,r)}$ denotes the theoretical upper bounds in [22] given as $\delta_{[\mathrm{EGL}^+20]}^{(d,r)} = \lfloor \log_2(d^r + 1) \rfloor$. We also verified our bounds on small-scale instances and the observed degree is denoted by $\delta^{(d,r)}$.

We observe that for $\mathsf{MiMC}_d(129, r)$ with $d \in \{3, 7, 15, 31\}$, the higher-order differential distinguisher can be established for up to $81, 46, 33, 27$ rounds, respectively. However, according to the formula for the number of rounds used in

MiMC specification [3], the total rounds are $82, 46, 34, 27$ rounds, respectively. The distinguishers even can cover the full-round $\mathsf{MiMC_d}$ for $d = 7$ and 31 while the security margin is only 1 round for $d = 3$ and 15. Therefore, it invalidates the security claims of the designers and we expect that more rounds than previously predicted in MiMC-like schemes are necessary to guarantee the security against the higher-order differential distinguisher.

We also investigate the algebraic degree of $\mathsf{MiMC}_d(n, r)$ with $d = 2^l + 1$. Besides the theoretical bound $\delta^{(d,r)}_{[\mathrm{EGL+20}]}$, the work of [14] proposed another theoretical bound for $d = 2^l + 1$, represented by $\delta^{(d,r)}_{[\mathrm{BCP22}]}$. $\delta^{(d,r)}_{[\mathrm{BCP22}]}$ [5] is given as

$$\delta^{(d,r)}_{[\mathrm{BCP22}]} = \begin{cases} 2 \times \lceil k_r/2 - 1 \rceil, k_r = \lfloor r \log_2(d) \rfloor, & \text{for } l = 1, \\ \lfloor r \log_2(d) \rfloor - l + 1, & \text{for } l > 1. \end{cases}$$

When $d = 3$, $\delta^{(3,r)}_{[\mathrm{BCP22}]}$ is exact for up to more than 16000 rounds of MiMC.

Appendix A in [1] compare the different upper bounds of the algebraic degree for $\mathsf{MiMC}_d(129, r)$ for exponents $d = 2^l + 1$. When $d = 3$, our detected bound seems to coincide with $\delta^{(3,r)}_{[\mathrm{BCP22}]}$, the exact algebraic degree. However, with the increase of d, the theoretical bounds $\delta^{(d,r)}_{[\mathrm{EGL+20}]}$ and $\delta^{(d,r)}_{[\mathrm{BCP22}]}$ do not match the observed bound $\delta^{(d,r)}$ as well as $d = 3$, even if the weight of d is the same. Instead, the upper bound found by our algorithm seems to coincide to the observed degree well. Overall, our bounds provide a more precise evaluation of the algebraic degree. This leads to the full-round or almost full-round higher-order differential distinguishers for different instances of $\mathsf{MiMC_d}$. All the results are summarized in Table 3.

5.3 Application to GMiMC

In this subsection, we focus on $\mathrm{GMiMC}_{\mathsf{erf}}$, which achieves the best performance among all the variants of GMiMC and has been chosen in the StarkWare challenges. We denote $\mathrm{GMiMC}_{\mathsf{erf}}$ specified by branch number t and block size n as $\mathsf{GMiMC}(n, t)$ for simplicity. It is an r-round block cipher in unbalanced Feistel network with an expanding round function, defined as

$$(x_0^{(i)}, x_1^{(i)}, \cdots, x_{t-1}^{(i)}) \leftarrow (x_1^{(i-1)} \oplus F(x_0^{(i-1)}), \cdots, x_{t-1}^{(i-1)} \oplus F(x_0^{(i-1)}), x_0^{(i-1)}),$$

where $x_j^{(i)}$ denotes the input of the j-th branch for round i. F represents the cubic mapping over finite field as

$$F(x) := (x \oplus k^{(i-1)})^3.$$

$k = (k^{(0)}, \cdots, k^{(r-1)})$ is a sequence of round keys and we omit the round constants for simplicity. The overall round function of $\mathsf{GMiMC}(n, t)$ is illustrated in Figure 4.

[5] [14] also gives an improved bound when $d \neq 3$. However, the cost for computing the Hamming weight is exponential in r, which means that the bound is infeasible to be determined computationally.

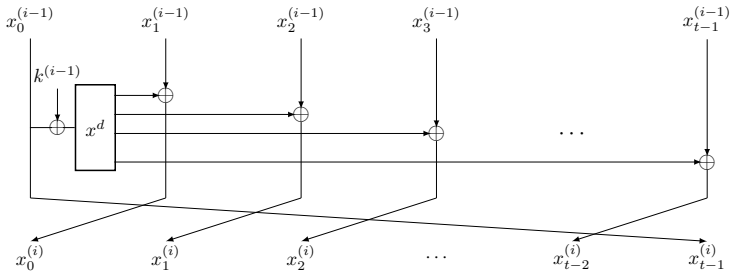

Fig. 4: The round function of $\mathsf{GMiMC}(n, t)$.

Higher-order differential distinguisher for $\mathbf{GMiMC_{erf}}$. With Model 4, we can apply Algorithm 1 to search for the higher-order differential distinguisher by slightly modifying the initial constraints as

$$\sum_{i \in \mathbb{X}} wt(u_i^{(0)}) = l.$$

$\mathbb{X}$ denotes the set of branches we focus on, For a concrete example, we search for the degree growth of $\mathsf{GMiMC}(33, 8)$.

By saturating three branches (x_5, x_6, x_7) of $\mathsf{GMiMC}(33, 8)$, our algorithm finds zero-sum in all the output variables after 29 rounds. Due to the structure of the unbalanced Feistel structure, we can extend the distinguishers for $t - 1$ more rounds according to Proposition 3 in [8], as a total of 36 rounds. Moreover, we can modify the initial constraint as

$$\sum_{i=0}^{t-1} wt(u_i^{(0)}) = t \cdot n - 1$$

and search for the longest higher-order differential distinguisher. If the model is infeasible, then the corresponding algebraic degree is strictly less than $n \cdot t - 1$ and we can always construct the distinguisher with the largest non-trivial vector space. The longest distinguisher we can find covers a total of 43 rounds with all the output branches achieving zero-sum property, which can be naturally extended to 50 rounds in the same way as before. It is 10 rounds longer than the distinguisher for permutation $\mathsf{GMiMC}(33, 8)$ found in [8].

6 Conclusion

While the traditional block ciphers defined over $\mathbb{F}_2$ possess a far-developed analysis toolbox, there is a lack of cryptanalytic methods for the novel arithmetization-oriented ciphers due to the quite different design constraints. In this paper, we introduce a novel extension of the division property, called *general monomial prediction*. It is a generic technique to detect the algebraic degree for ciphers

over $\mathbb{F}_{2^n}$ by evaluating whether the polynomial representation of a block cipher contains some specific monomials. Through tracing the transition of the exponents, we develop a searching tool for the degree estimation of ciphers based on the relationship between the exponents of monomials and the algebraic degree. We apply our algorithm to some competitive arithmetization-oriented block ciphers including MiMC, Feistel MiMC, and GMiMC. As a result, we successfully find the currently best degree bounds and get much longer distinguishers than previous results for several instances. Overall, our methods provide a better estimation for the algebraic degree in case of ciphers over the finite field $\mathbb{F}_{2^n}$ and furthermore, help to establish a more accurate number of rounds necessary to guarantee the security level.

Acknowledgements. We thank the anonymous reviewers for their valuable comments. This work is supported by the National Natural Science Foundation of China (Grant No. 62032014), the National Key Research and Development Program of China (Grant No. 2018YFA0704702), the Major Basic Research Project of Natural Science Foundation of Shandong Province, China (Grant No. ZR202010220025). Kai Hu is supported by the "ANR-NRF project SELECT". Puwen Wei is supported by the Shandong Provincial Natural Science Foundation (No. ZR2020MF053).

References

1. https://eprint.iacr.org/2022/1210.pdf.
2. M. R. Albrecht, L. Grassi, L. Perrin, S. Ramacher, C.n Rechberger, D. Rotaru, A. Roy, and M. Schofnegger. Feistel structures for mpc, and more. In *ESORICS 2019*, volume 11736 of *Lecture Notes in Computer Science*, pages 151–171. Springer, 2019.
3. M. R. Albrecht, L. Grassi, C. Rechberger, A. Roy, and T. Tiessen. Mimc: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. In *ASIACRYPT 2016*, volume 10031 of *Lecture Notes in Computer Science*, pages 191–219, 2016.
4. A. Aly, T. Ashur, E. Ben-Sasson, S. Dhooghe, and A. Szepieniec. Design of symmetric-key primitives for advanced cryptographic protocols. *IACR Trans. Symmetric Cryptol.*, 2020(3):1–45, 2020.
5. Jean-Philippe Aumasson and Willi Meier. Zero-sum distinguishers for reduced keccak-f and for the core functions of luffa and hamsi. *rump session of Cryptographic Hardware and Embedded Systems-CHES*, 2009:67, 2009.
6. C. Barrett, R. Sebastiani, S. A. Seshia, and C. Tinelli. Satisfiability modulo theories. In *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 825–885. IOS Press, 2009.
7. C. Barrett and C. Tinelli. CVC3. In *CAV 2007*, volume 4590 of *Lecture Notes in Computer Science*, pages 298–302. Springer, 2007.
8. T. Beyne, A. Canteaut, I. Dinur, M. Eichlseder, G. Leander, G. Leurent, M. Naya-Plasencia, L. Perrin, Y. Sasaki, Y. Todo, and F. Wiemer. Out of oddity - new cryptanalytic techniques against symmetric primitives optimized for integrity proof systems. In *CRYPTO 2020*, volume 12172 of *Lecture Notes in Computer Science*, pages 299–328. Springer, 2020.

9. E. Biham and A. Shamir. Differential cryptanalysis of des-like cryptosystems. In *CRYPTO '90*, volume 537 of *Lecture Notes in Computer Science*, pages 2–21. Springer, 1990.
10. C. Boura and A. Canteaut. On the influence of the algebraic degree of f^{-1} on the algebraic degree of G ∘ F. *IEEE Trans. Inf. Theory*, 59(1):691–702, 2013.
11. C. Boura and A. Canteaut. Another view of the division property. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 654–682. Springer, 2016.
12. C. Boura, A. Canteaut, and C. Cannière. Higher-order differential properties of keccak and *Luffa*. In *FSE 2011*, volume 6733 of *Lecture Notes in Computer Science*, pages 252–269. Springer, 2011.
13. C. Boura and D. Coggia. Efficient MILP modelings for sboxes and linear layers of SPN ciphers. *IACR Trans. Symmetric Cryptol.*, 2020(3):327–361, 2020.
14. C. Bouvier, A. Canteaut, and L. Perrin. On the algebraic degree of iterated power functions. Cryptology ePrint Archive, Report 2022/366, 2022. https://ia.cr/2022/366.
15. A. Canteaut and M. Videau. Degree of composition of highly nonlinear functions and applications to higher order differential cryptanalysis. In *EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 518–533. Springer, 2002.
16. C. Carlet, P. Charpin, and V. A. Zinoviev. Codes, bent functions and permutations suitable for des-like cryptosystems. *Des. Codes Cryptogr.*, 15(2):125–156, 1998.
17. S. Chen, Z. Xiang, X. Zeng, and S. Zhang. On the relationships between different methods for degree evaluation. *IACR Trans. Symmetric Cryptol.*, 2021(1):411–442, 2021.
18. C. Cid, L. Grassi, A. Gunsing, R. Lüftenegger, C. Rechberger, and M. Schofnegger. Influence of the linear layer on the algebraic degree in sp-networks. *IACR Trans. Symmetric Cryptol.*, 2022(1):110–137, 2022.
19. S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing, 1971*, pages 151–158. ACM, 1971.
20. P. Derbez and P. Fouque. Increasing precision of division property. *IACR Trans. Symmetric Cryptol.*, 2020(4):173–194, 2020.
21. C. Dobraunig, L. Grassi, A. Guinet, and D. Kuijsters. Ciminion: Symmetric encryption based on toffoli-gates over large finite fields. In *EUROCRYPT 2021*, volume 12697 of *Lecture Notes in Computer Science*, pages 3–34. Springer, 2021.
22. M. Eichlseder, L. Grassi, R. Lüftenegger, Mo. Øygarden, C. Rechberger, M. Schofnegger, and Q. Wang. An algebraic attack on ciphers with low-degree round functions: Application to full mimc. In *ASIACRYPT 2020*, volume 12491 of *Lecture Notes in Computer Science*, pages 477–506. Springer, 2020.
23. V. Ganesh and D. L. Dill. A decision procedure for bit-vectors and arrays. In *CAV 2007*, volume 4590 of *Lecture Notes in Computer Science*, pages 519–531. Springer, 2007.
24. L. Grassi, R. Lüftenegger, C. Rechberger, D. Rotaru, and M. Schofnegger. On a generalization of substitution-permutation networks: The HADES design strategy. In *EUROCRYPT 2020*, volume 12106 of *Lecture Notes in Computer Science*, pages 674–704. Springer, 2020.
25. Y. Hao, G., W. Meier, Y. Todo, and Q. Wang. Modeling for three-subset division property without unknown subset - improved cube attacks against trivium and grain-128aead. In *EUROCRYPT 2020*, volume 12105 of *Lecture Notes in Computer Science*, pages 466–495. Springer, 2020.

26. P. Hebborn, B. Lambin, G. Leander, and Todo Y. Lower bounds on the degree of block ciphers. In *ASIACRYPT 2020*, volume 12491 of *Lecture Notes in Computer Science*, pages 537–566. Springer, 2020.
27. K. Hu, S. Sun, M. Wang, and Q. Wang. An algebraic formulation of the division property: Revisiting degree evaluations, cube attacks, and key-independent sums. In *ASIACRYPT 2020*, volume 12491 of *Lecture Notes in Computer Science*, pages 446–476. Springer, 2020.
28. L. Knudsen and V. Rijmen. Known-key distinguishers for some block ciphers. In *ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 315–324. Springer, 2007.
29. X. Lai. Higher order derivatives and differential cryptanalysis. In *Communications and cryptography*, pages 227–233. Springer, 1994.
30. B. Lambin, P. Derbez, and P. Fouque. Linearly equivalent s-boxes and the division property. *Des. Codes Cryptogr.*, 88(10):2207–2231, 2020.
31. M. Matsui. Linear cryptanalysis method for DES cipher. In *EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer, 1993.
32. M. Soos, K. Nohl, and C. Castelluccia. Extending SAT solvers to cryptographic problems. In *SAT 2009*, volume 5584 of *Lecture Notes in Computer Science*, pages 244–257. Springer, 2009.
33. L. Sun, W. Wang, and M. Wang. Automatic search of bit-based division property for ARX ciphers and word-based division property. In *ASIACRYPT 2017*, volume 10624 of *Lecture Notes in Computer Science*, pages 128–157. Springer, 2017.
34. Y. Todo. Structural evaluation by generalized integral property. In *EUROCRYPT 2015*, volume 9056 of *Lecture Notes in Computer Science*, pages 287–314. Springer, 2015.
35. Y. Todo and M. Morii. Bit-based division property and application to simon family. In *FSE 2016*, volume 9783 of *Lecture Notes in Computer Science*, pages 357–377. Springer, 2016.
36. A. Udovenko. Convexity of division property transitions: Theory, algorithms and compact models. In *ASIACRYPT 2021*, volume 13090 of *Lecture Notes in Computer Science*, pages 332–361. Springer, 2021.
37. S. Wang, B. Hu, J. Guan, K. Zhang, and T. Shi. Milp-aided method of searching division property using three subsets and applications. In *ASIACRYPT 2019*, volume 11923 of *Lecture Notes in Computer Science*, pages 398–427. Springer, 2019.
38. Z. Xiang, W. Zhang, Z. Bao, and D. Lin. Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers. In *ASIACRYPT 2016*, volume 10031 of *Lecture Notes in Computer Science*, pages 648–678, 2016.