Automatic Search of Bit-Based Division Property for ARX Ciphers and Word-Based Division Property

Ling Sun[1,2], Wei Wang[1], Meiqin Wang*[1,2,3]

[1] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan, 250100, China
[2] Science and Technology on Communication Security Laboratory, Chengdu 610041, China
[3] State Key Laboratory of Cryptology, P.O. Box 5159, Beijing, 100878, China
lingsun@mail.sdu.edu.cn; weiwangsdu@sdu.edu.cn; mqwang@sdu.edu.cn

Abstract. Division property is a generalized integral property proposed by Todo at Eurocrypt 2015. Previous tools for automatic searching are mainly based on the Mixed Integer Linear Programming (MILP) method and trace the division property propagation at the bit level. In this paper, we propose automatic tools to detect ARX ciphers' division property at the bit level and some specific ciphers' division property at the word level.

For ARX ciphers, we construct the automatic searching tool relying on Boolean Satisfiability Problem (SAT) instead of MILP, since SAT method is more suitable in the search of ARX ciphers' differential/linear characteristics. The propagation of division property is translated into a system of logical equations in Conjunctive Normal Form (CNF). Some logical equations can be dynamically adjusted according to different initial division properties and stopping rule, while the others corresponding to r-round propagations remain the same. Moreover, our approach can efficiently identify some optimized distinguishers with lower data complexity. As a result, we obtain a 17-round distinguisher for SHACAL-2, which gains four more rounds than previous work, and an 8-round distinguisher for LEA, which covers one more round than the former one.

For word-based division property, we develop the automatic search based on Satisfiability Modulo Theories (SMT), which is a generalization of SAT. We model division property propagations of basic operations and S-boxes by logical formulas, and turn the searching problem into an SMT problem. With some available solvers, we achieve some new distinguishers. For CLEFIA, 10-round distinguishers are obtained, which cover one more round than the previous work. For the internal block cipher of Whirlpool, the data complexities of 4/5-round distinguishers are improved. For Rijndael-192 and Rijndael-256, 6-round distinguishers are presented, which attain two more rounds than the published ones. Besides, the integral attacks for CLEFIA are improved by one round with the newly obtained distinguishers.

Keywords: Automatic search, Division property, ARX, SAT/SMT

* Corresponding Author

1 Introduction

Automatic tools for cryptanalysis play a more and more important role in the design and cryptanalysis of symmetric ciphers. One common direction to construct automatic tools is to transform the searching problems into some mathematical problems, so that some existing solvers can be invoked. The involved mathematical problems can be roughly divided into three categories, which are Boolean Satisfiability Problem (SAT)/Satisfiability Modulo Theories (SMT) problem [7,16,24,32], Mixed Integer Linear Programming (MILP) problem [10,25,39,45], and Constraint Programming (CP) problem [12,38]. At the very start, the researches on automatic search of distinguishers concentrated on detecting differential and linear characteristics, since differential [4] and linear [20] cryptanalysis are two of the most powerful techniques in cryptanalysis of symmetric-key primitives. Recently, with the advent of division property [41], which is a generalized integral property, some researches about automatic searching for division property arose.

Division property was proposed by Todo [41] at Eurocrypt 2015, which was originally used to search integral distinguishers of block cipher structures. Due to the newly identified division property, at Crypto 2015, MISTY1 [21] was broken by Todo for the first time. Later, Todo and Morii [42] introduced the bit-based division property at FSE 2016, which propagates each bit independently, and a 14-round integral distinguisher for SIMON32 [3] was detected. Depending on the partition of the internal state, the methods behind the obtained distinguishers can be divided into three categories. 1) *state-based* division property: evaluate the division properties of some generalized structures. Todo [41] finished the extensive research for 2-branch Feistel structure and SPN on the whole state. Related works were provided in [5]. 2) *word-based* division property: evaluate the division properties of some specific ciphers at the word level. Todo [41] implemented the search for a variety of AES-like ciphers with 4-bit S-boxes, and the 6-round integral distinguisher [40] for MISTY1 was obtained based on this method. Some works on this topic were introduced in [34, 35, 46]. 3) *bit-based* division property: evaluate the propagation of division property at the bit level. Note that it is more likely to obtain better distinguishers under a more subtle partition since more information can be taken into account. All published automatic tools of integral distinguishers based on division property focused on the bit level. At Asiacrypt 2016, Xiang et al. [45] applied MILP method to search integral distinguishers with bit-based division property. Soon after, the automatic search of integral distinguishers based on MILP method for ARX ciphers was proposed in [36]. Many other automatic tools relying on MILP and CP can be found in [37,38,47].

ARX ciphers constitute a broad class of symmetric-key cryptographic algorithms, and are composed of a small set of simple operations such as modular addition, bit rotation, bit shift and `XOR`. To claim the security of ARX ciphers, one way is to prove the security bounds just as Dinu et al. showed in [9], where a long trail design strategy for ARX ciphers with provable bounds was proposed. The other is to estimate the maximum number of rounds of the detectable distin-

guishers which heavily relied on automatic tools, and the searching of distinguishers is converted into an SAT/SMT problem or MILP problem. The results show that SAT/SMT based methods [18, 24, 32] outperform MILP based methods [10] in the search of differential/linear characteristics for ARX ciphers. Hence, for bit-based division property, it is worth exploring whether automatic tools based on SAT/SMT method can be constructed and provide better performance for ARX primitives.

Although the search of bit-based division property can take advantage of more details, it is infeasible to trace the division property propagation at the bit level for some ciphers with large state and complicated operations, such as Rijndael [8] with 256-bit block size. In order to get the tradeoff between accuracy and practicability as we detect the division property, we also consider building automatic tool to search integral distinguishers on account of word-based division property.

Our Contributions. For the integral cryptanalysis, we construct automatic searching tools of bit-based division property for ARX ciphers and word-based division property for some specific ciphers. The key point is to translate the propagation of division property into an SAT/SMT problem and control the function calls. Specifically, the contributions can be summarized as follows:

- For ARX ciphers, we propose automatic tools to search integral distinguishers using bit-based division property. First, we model the division property propagations of the three basic operations, i.e., **Copy**, `AND`, and `XOR`, and present formulas in Conjunctive Normal Form (CNF) for them. Then, the concrete equations for the modular addition operation to depict bit-based division property propagation can be achieved. The initial division property and stopping rule are transformed to logical equations, too. At last, the propagation of division property for ARX cipher is described by a system of logical equations in CNF, where some logical formulas can be dynamically adjusted according to different initial division properties of the input multi-set and final division properties of the output multi-set, and the others corresponding to r-round propagations remain the same.
- For integral cryptanalysis, it is better to adopt distinguishers with less data requirements, and our approach can efficiently identify some optimal[4] distinguishers which require less chosen plaintexts among the distinguishers with the same length. Our searching approach is composed of two algorithms. The first one restricts the search scope of initial division property and determines the maximum number of rounds of distinguishers achieved in our model. The second one optimizes the distinguishers based on the first algorithm's output.
- For word-based division property, we construct automatic tool based on SMT method. We first study how to model division property propagations of basic operations by logical formulas. Moreover, by exclusion method, we construct

[4] The integral distinguishers are optimal under the search strategies defined in this paper.

formulas to depict the possible propagations calculated by the Substitution rule. With some available solvers, we can efficiently search integral distinguishers by setting initial division property and stopping rule rationally. Finally, the problem of searching division property can be transformed into an SMT problem.

- New integral distinguishers are detected for some ARX ciphers, such as SHACAL-2 [13], LEA [14], and HIGHT [15]. With the two algorithms mentioned above, the number of initial division properties required to be evaluated for SHACAL-2 is reduced from $2^{79.24}$ to 410, so that we can easily obtain a 17-round integral distinguisher with data complexity 2^{241} chosen plaintexts, which achieves four more rounds than previous work. For LEA, an 8-round distinguisher is identified, which covers one more round than the one found by MILP method [36]. For HIGHT, although the lengths and data requirements of the newly obtained distinguishers are not improved, some of them have more zero-sum bits than those proposed in [36].
- New word-based division properties are presented for some specific ciphers. For CLEFIA [31], we discover 10-round distinguishers, which attain one more round than the one proposed in [19]. With the newly obtained distinguishers for CLEFIA, we can improve the previous integral attacks by one round. The data requirements of 4/5-round integral distinguishers for the internal block cipher of Whirlpool [1] are reduced. As to Rijndael-192 and Rijndael-256 [8], 6-round distinguishers are proposed, which cover two more rounds than the previous work.

Our main results and the comparisons are listed in Table 1 and Table 3.

The rest of the paper is organized as follows. In Section 2, some notations and background are introduced. Section 3 focuses on the automatic search of integral distinguishers with bit-based division property for ARX ciphers. The automatic method relying on SMT to search integral distinguishers in accordance with word-based division property is provided in Section 4. Section 5 presents some applications of the developed automatic tools. We conclude the paper in Section 6.

2 Preliminary

2.1 Notations

For any $a \in \mathbb{F}_2^n$, its i-th element is denoted as $a[i]$, where the bit positions are labeled in big-endian, and the Hamming weight $w(a)$ is calculated by $w(a) = \sum_{i=0}^{n-1} a[i]$. For any $\boldsymbol{a} = (a_0, a_1, \ldots, a_{m-1}) \in \mathbb{F}_2^{\ell_0} \times \mathbb{F}_2^{\ell_1} \times \cdots \times \mathbb{F}_2^{\ell_{m-1}}$, the vectorial Hamming weight of $\boldsymbol{a}$ is defined as $W(\boldsymbol{a}) = (w(a_0), w(a_1), \ldots, w(a_{m-1})) \in \mathbb{Z}^m$. For any $\boldsymbol{k} \in \mathbb{Z}^m$ and $\boldsymbol{k'} \in \mathbb{Z}^m$, we define $\boldsymbol{k} \succeq \boldsymbol{k'}$ if $k_i \geq k'_i$ for all i. Otherwise, $\boldsymbol{k} \not\succeq \boldsymbol{k'}$.

For any set $\mathbb{K}$, $|\mathbb{K}|$ denotes the number of elements in $\mathbb{K}$. $\emptyset$ stands for an empty set. Denote $\mathbb{Z}_m$ the set $\{0, 1, \ldots, m\}$.

Definition 1 (Bit Product Function). *Assume $u \in \mathbb{F}_2^n$ and $x \in \mathbb{F}_2^n$. The Bit Product Function π_u is defined as*

$$\pi_u(x) = \prod_{i=0}^{n-1} x[i]^{u[i]}.$$

For $\boldsymbol{u} = (u_0, u_1, \ldots, u_{m-1}) \in \mathbb{F}_2^{\ell_0} \times \mathbb{F}_2^{\ell_1} \times \cdots \times \mathbb{F}_2^{\ell_{m-1}}$, let $\boldsymbol{x} = (x_0, x_1, \ldots, x_{m-1}) \in \mathbb{F}_2^{\ell_0} \times \mathbb{F}_2^{\ell_1} \times \cdots \times \mathbb{F}_2^{\ell_{m-1}}$ be the input, the Bit Product Function $\pi_{\boldsymbol{u}}$ is defined as

$$\pi_{\boldsymbol{u}}(\boldsymbol{x}) = \prod_{i=0}^{m-1} \pi_{u_i}(x_i).$$

2.2 Division Property

The original integral distinguishers mainly focus on the propagation of ALL and BALANCE properties [17]. While, the division property, proposed by Todo at Eurocrypt 2015 [41], is a generalized integral property, which traces the implicit properties between traditional ALL and BALANCE properties. First, a set of plaintexts, whose division property follows initial division property, is chosen. Then, the division property of the set of texts encrypted over one round is deduced from the propagation rules. And so on, we can exploit the division property over several rounds, and determine the existence of the integral distinguishers. In the following, we briefly recall the definition of division property, and propagation rules for basic operations involved in the encryption process.

Definition 2 (Division Property [41]). *Let $\mathbb{X}$ be a multi-set whose elements take values from $\mathbb{F}_2^{\ell_0} \times \mathbb{F}_2^{\ell_1} \times \cdots \times \mathbb{F}_2^{\ell_{m-1}}$. When the multi-set $\mathbb{X}$ has the division property $\mathcal{D}_{\mathbb{K}}^{\ell_0,\ell_1,\ldots,\ell_{m-1}}$, where $\mathbb{K}$ denotes a set of m-dimensional vectors whose i-th element takes a value between 0 and ℓ_i, it fulfills the following conditions:*

$$\bigoplus_{\boldsymbol{x}\in\mathbb{X}} \pi_{\boldsymbol{u}}(\boldsymbol{x}) = \begin{cases} \textit{unknown} & \textit{if there is } \boldsymbol{k} \in \mathbb{K} \textit{ s.t. } W(\boldsymbol{u}) \succeq \boldsymbol{k}, \\ 0 & \textit{otherwise.} \end{cases}$$

Remark 1. Note that ℓ_0, ℓ_1, …, ℓ_{m-1} are restricted to 1 when we consider **bit-based division property**.

Propagation Rules for Division Property.

Rule 1 (Substitution [41]) *Let F be a function that consists of m S-boxes, where the bit length and the algebraic degree of the i-th S-box is ℓ_i and d_i bits, respectively. The input and the output take values from $\mathbb{F}_2^{\ell_0} \times \mathbb{F}_2^{\ell_1} \times \cdots \times \mathbb{F}_2^{\ell_{m-1}}$, and $\mathbb{X}$ and $\mathbb{Y}$ denote the input and output multi-sets, respectively. Assuming that $\mathbb{X}$ has division property $\mathcal{D}_{\mathbb{K}}^{\ell_0,\ell_1,\ldots,\ell_{m-1}}$, where $\mathbb{K}$ denotes a set of m-dimensional*

Table 1: Summary of Integral Distinguishers.

Cipher	Block Size	Key Size	Round†	Length‡	$\log_2$(Data)	Balanced Bits	Reference
SHACAL-2	256	128 ∼ 512	64	12	1	32	[43]
				13	32	1	[30]
				17	241	7	Section 5.1
LEA	128	128/192/256	24/28/32	6	32	1	[14]
				6	32	2	[36]
				7	96	1	[36]
				8	118	1	Section 5.1
CLEFIA	128	128/192/256	18/22/26	6	32	32	[31]
				8	96	32	[31]
				9	112	32	[19]
				9	105	24	[29]
				10	127	64	Section 5.2
Rijndael	192	128/192/256	12/12/14	4	24	160	[23]
				4	176	192	[41]
				6	160	64	Section 5.2
	256	128/192/256	14	4	24	64	[11, 23]
				4	232	256	[41]
				6	160	64	Section 5.2
Whirlpool	512	-	10	4	64	512	[22]
				5	488	512	[41]
				5	384	512	Section 5.2

† the number of encryption rounds.
‡ the number of rounds covered by the distinguisher.

vectors whose i-th element takes a value between 0 and ℓ_i, the division property of $\mathbb{Y}$ is $\mathcal{D}^{\ell_0,\ell_1,\ldots,\ell_{m-1}}_{\mathbb{K}'}$, where[5]

$$\mathbb{K}' = \left\{ \left(\left\lceil \frac{k_0}{d_0} \right\rceil, \left\lceil \frac{k_1}{d_1} \right\rceil, \cdots, \left\lceil \frac{k_{m-1}}{d_{m-1}} \right\rceil \right) \middle| \boldsymbol{k} = (k_0, k_1, \ldots, k_{m-1}) \in \mathbb{K} \right\}.$$

Rule 2 (Copy [41]) *Let F be a copy function, where the input x takes value from $\mathbb{F}_2^n$ and the output is calculated as $(y_0, y_1) = (x, x)$. Let $\mathbb{X}$ and $\mathbb{Y}$ be the input and output multi-sets, respectively. Assuming that $\mathbb{X}$ has the division property $\mathcal{D}^n_{\{k\}}$, the division property of $\mathbb{Y}$ is $\mathcal{D}^{n,n}_{\mathbb{K}'}$, where*

$$\mathbb{K}' = \{(k-i, i) | 0 \leqslant i \leqslant k\}.$$

[5] The same vector is not inserted twice, similarly hereinafter.

Rule 3 (XOR [41]) *Let F be an XOR function, where the input (x_0, x_1) takes value from $\mathbb{F}_2^n \times \mathbb{F}_2^n$ and the output is calculated as $y = x_0 \oplus x_1$. Let $\mathbb{X}$ and $\mathbb{Y}$ be the input and output multi-sets, respectively. Assuming that $\mathbb{X}$ has division property $\mathcal{D}_{\mathbb{K}}^{n,n}$, the division property of $\mathbb{Y}$ is $\mathcal{D}_{\{k'\}}^{n}$, where*

$$k' = \min\{k_0 + k_1 | (k_0, k_1) \in \mathbb{K}\}.$$

Here, if k' is larger than n, the propagation characteristic of division property is aborted. Namely, a value of $\bigoplus_{y \in \mathbb{Y}} \pi_v(y)$ is 0 for all $v \in \mathbb{F}_2^n$.

Rule 4 (Split [41]) *Let F be a split function, where the input x is an element belonging to $\mathbb{F}_2^n$ and the output is calculated as $y_0 \| y_1 = x$, where (y_0, y_1) takes value from $\mathbb{F}_2^{n_0} \times \mathbb{F}_2^{n-n_0}$. Let $\mathbb{X}$ and $\mathbb{Y}$ be the input and output multi-sets, respectively. Assuming that $\mathbb{X}$ has the division property $\mathcal{D}_{\{k\}}^{n}$, the division property of $\mathbb{Y}$ is $\mathcal{D}_{\mathbb{K}'}^{n_0, n-n_0}$, where*

$$\mathbb{K}' = \{(k-i, i) | 0 \leqslant i \leqslant k, k-i \leqslant n_0, i \leqslant n-n_0\}.$$

Rule 5 (Concatenation [41]) *Let F be a concatenation operation, where the input (x_0, x_1) takes value from $\mathbb{F}_2^{n_0} \times \mathbb{F}_2^{n_1}$ and the output is calculated as $y = x_0 \| x_1$. Let $\mathbb{X}$ and $\mathbb{Y}$ be the input and output multi-sets, respectively. Assuming that $\mathbb{X}$ has the division property $\mathcal{D}_{\mathbb{K}}^{n_0, n_1}$, the division property of $\mathbb{Y}$ is $\mathcal{D}_{\{k'\}}^{n_0+n_1}$, where*

$$k' = \min\{k_0 + k_1 | (k_0, k_1) \in \mathbb{K}\}.$$

The above rules are defined at the word level, while, when it comes to bit-based division property, **Copy** and XOR rules can be applied, naturally. Another important propagation rule under bit-based division property is AND, which is stated in the following.

Rule 6 (Bit-based AND [42]) *Let F be an AND function, where the input (x_0, x_1) takes value from $\mathbb{F}_2 \times \mathbb{F}_2$, and the output is calculated as $y = x_0 \wedge x_1$. Let $\mathbb{X}$ and $\mathbb{Y}$ be the input and output multi-sets, respectively. Assuming that $\mathbb{X}$ has division property $\mathcal{D}_{\mathbb{K}}^{1,1}$, the division property of $\mathbb{Y}$ is $\mathcal{D}_{\mathbb{K}'}^{1}$, where*

$$\mathbb{K}' = \left\{ \left\lceil \frac{k_0 + k_1}{2} \right\rceil \middle| \boldsymbol{k} = (k_0, k_1) \in \mathbb{K} \right\}.$$

Similar to differential/linear characteristic in differential/linear cryptanalysis, the concatenation of r division properties of the internal states constitutes an r-round division trail, which is formally defined in the following.

Definition 3 (Division Trail [45]). *Let f be the round function of an iterated block cipher. Assume that the input multi-set has division property $\mathcal{D}_{\{\boldsymbol{k}\}}^{\ell_0, \ell_1, \cdots, \ell_{m-1}}$,*

and the internal state after i rounds has division property $\mathcal{D}_{\mathbb{K}_i}^{\ell_0,\ell_1,\cdots,\ell_{m-1}}$. Thus we have the following chain of division property propagations:

$$\{\boldsymbol{k}\} \triangleq \mathbb{K}_0 \xrightarrow{f} \mathbb{K}_1 \xrightarrow{f} \mathbb{K}_2 \xrightarrow{f} \cdots \xrightarrow{f} \mathbb{K}_r.$$

Moreover, for any vector $\boldsymbol{k}_i^* \in \mathbb{K}_i$ $(i \geqslant 1)$, there must exist a vector $\boldsymbol{k}_{i-1}^* \in \mathbb{K}_{i-1}$ such that $\boldsymbol{k}_{i-1}^*$ can propagate to $\boldsymbol{k}_i^*$ by propagation rules. Furthermore, for $(\boldsymbol{k}_0, \boldsymbol{k}_1, \ldots, \boldsymbol{k}_r) \in \mathbb{K}_0 \times \mathbb{K}_1 \times \cdots \times \mathbb{K}_r$, if $\boldsymbol{k}_{i-1}$ can propagate to $\boldsymbol{k}_i$ for all $i \in \{1, 2, \ldots, r\}$, we call $(\boldsymbol{k}_0, \boldsymbol{k}_1, \ldots, \boldsymbol{k}_r)$ an r-round division trail.

The propagation of division property round by round will eventually lead to a multi-set without integral property. The following proposition can be used to detect whether a set has integral property or not, which helps us to decide when to stop propagating.

Proposition 1 (Set without Integral Property [45]). *Assume $\mathbb{X}$ is a multi-set satisfying division property $\mathcal{D}_{\mathbb{K}}^{\ell_0,\ell_1,\cdots,\ell_{m-1}}$, then $\mathbb{X}$ does not have integral property if and only if $\mathbb{K}$ contains all vectors with vectorial Hamming weight 1.*

Distinguishing Attacks with Division Property.

Suppose the output division property of an integral distinguisher has balanced property on b bits. Once the sum for each of the b bits is zero, the distinguisher $\mathcal{D}$ outputs '1'; otherwise, outputs '0'. The success rate of the distinguishing attack p is composed of two cases: one is $\mathcal{D}$ outputs '1' when the oracle $\mathcal{O}$ is a concrete cipher $\mathcal{F}$ actually, the other is $\mathcal{D}$ outputs '0' when $\mathcal{O}$ is a random permutation $\mathcal{RP}$. For $\mathcal{F}$, the balanced property holds with probability 1, while for $\mathcal{RP}$ is 2^{-b}. Assuming that the probability of whether the oracle is $\mathcal{F}$ or $\mathcal{RP}$ is 0.5, it is clear that $p = 0.5 \cdot 1 + 0.5 \cdot (1 - 2^{-b}) = 1 - 2^{-b-1}$, which is 0.75 for $b = 1$, and is count for distinguishing attack.

In order to increase the success rate, we can repeat the distinguishing attack with different chosen-plaintext structures. For an n-bit cipher, suppose that the input division property requires that t bits need to be traversed. Then, the number of times the distinguishers can be replayed is at most 2^{n-t}. The data complexity of the distinguishing attack need to be discussed accordingly.

2.3 SAT & SMT Problems

In computer science, the Boolean Satisfiability Problem (SAT) [6] is the problem of determining if there exists an interpretation that satisfies a given Boolean formula. In other words, it discusses whether the variables involved in a given Boolean formula can be consistently replaced by the value `True` or `False` so that the formula is evaluated to be `True`. If this is the case, the formula is called *satisfiable*.

The Satisfiability Modulo Theories (SMT) [2] problem is a decision problem for logical formulas expressed in classical first-order logic with equality. An SMT instance is a generalization of SAT instance in which various sets of variables

are replaced by predicates from a variety of underlying theories. SMT formulas provide a much richer modeling language than is possible with SAT formulas.

To solve SAT and SMT problems, there are many openly available solvers, and we use CryptoMiniSat[6] and STP[7], respectively. In order to search integral distinguishers efficiently, we adopt the C++ interface of CryptoMiniSat and the Python interface of STP.

3 Automatic Search of Bit-Based Division Property for ARX Ciphers

For ARX ciphers, since SAT/SMT method [18, 24, 32] is more suitable to search for differential/linear characteristics than MILP method [10], we construct the automatic searching tool relying on SAT instead of MILP. First, we model the division property propagations of three basic operations, i.e., **Copy**, `AND`, and `XOR`, and construct formulas in Conjunctive Normal Form (CNF) for them. Then, the model used to describe bit-based division property propagation for the modular addition operation is constructed based on the three basic models. By setting initial division property and stopping rule appropriately, the problem of searching integral distinguishers using bit-based division property for ARX ciphers can be converted into an SAT problem, and settled efficiently.

3.1 Models of Basic Operations at the Bit Level

We consider the division property propagations of the three basic operations (**Copy**, `AND` and `XOR`) at the bit level, and the input and output are composed of bit variables which take a value of 0 or 1. Then the division trails of each operation correspond to vectors formed by the input and output variables. To depict the propagations of these operations, we translate the rules in Section 2.2 into formulas in CNF, of which the solutions correspond to all the possible division trails. More specifically, we first determine all the vectors corresponding to division trails, and then exclude those impossible vector values by logical formulas. We call this idea the *exclusion method*. By analyzing all the possible division trails of bit-based **Copy**, `AND` and `XOR` operations, we construct models to describe bit-based division property propagations for them.

Model 1 (Bit-based Copy) *Denote* $(a) \xrightarrow{\boldsymbol{Copy}} (b_0, b_1)$ *a division trail of* ***Copy*** *operation, the following logical equations are sufficient to depict the propagation of bit-based division property,*

$$\begin{cases} \overline{b_0} \vee \overline{b_1} = 1 \\ a \vee b_0 \vee \overline{b_1} = 1 \\ a \vee \overline{b_0} \vee b_1 = 1 \\ \overline{a} \vee b_0 \vee b_1 = 1 \end{cases}.$$

[6] https://github.com/msoos/cryptominisat
[7] http://stp.github.io/

Proof: Let (a, b_0, b_1) be the 3-bit vector composed of the input and output division properties. For an arbitrary 3-bit vector, it has eight possible values, which are

$$\textbf{(0, 0, 0)}, (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), \textbf{(1, 0, 1)}, \textbf{(1, 1, 0)}, (1, 1, 1).$$

When restricting to **Copy** operation, there are three division trails corresponding to the values in bold above. Thus, $(*, 1, 1)$, $(0, 0, 1)$, $(0, 1, 0)$, and $(1, 0, 0)$ are impossible cases required to be excluded, where $*$ can take 0 or 1.

In order to eliminate $(*, 1, 1)$, we assert $\overline{b_0} \vee \overline{b_1} = 1$. With this assertion, (a, b_0, b_1) cannot take values of the form $(*, 1, 1)$. Then, after eliminating all impossible cases in a similar way, we obtain the set of formulas in CNF to describe bit-based division property propagation of **Copy** operation. □

When it comes to bit-based `AND` operation, similar to the procedure for **Copy** operation, we consider all the possible division trails. Denote (a_0, a_1) the bit variables representing the input division property of `AND` operation, and let b be the bit variable standing for the output division property. Obviously, there are four division trails for `AND` operation, which are $(0, 0) \rightarrow (0)$, $(1, 0) \rightarrow (1)$, $(0, 1) \rightarrow (1)$, and $(1, 1) \rightarrow (1)$. Therefore, the set of logical equations have four solutions corresponding to (a_0, a_1, b), i.e., $(0, 0, 0)$, $(0, 1, 1)$, $(1, 0, 1)$, and $(1, 1, 1)$. Thus, we need to delete the impossible ones as follows.

Model 2 (Bit-based `AND`) *Denote* $(a_0, a_1) \xrightarrow{AND} (b)$ *a division trail of* `AND` *function, the following logical equations are sufficient to describe bit-based division property propagation of* `AND` *operation,*

$$\begin{cases} \overline{a_1} \vee b = 1 \\ a_0 \vee a_1 \vee \overline{b} = 1 \\ \overline{a_0} \vee b = 1 \end{cases}.$$

For bitwise `XOR` operation, only three division trails are possible, which are $(0, 0, 0)$, $(0, 1, 1)$, $(1, 0, 1)$, and the model can be constructed in a similar way.

Model 3 (Bit-based `XOR`) *Denote* $(a_0, a_1) \xrightarrow{XOR} (b)$ *a division trail of* `XOR` *function, the following logical equations are sufficient to evaluate the bit-based division property through* `XOR` *operation,*

$$\begin{cases} \overline{a_0} \vee \overline{a_1} = 1 \\ a_0 \vee a_1 \vee \overline{b} = 1 \\ a_0 \vee \overline{a_1} \vee b = 1 \\ \overline{a_0} \vee a_1 \vee b = 1 \end{cases}.$$

For specific ciphers, such as HIGHT [15], TEA [44], and XTEA [26], we also encounter cases where the number of output branches for **Copy** operation or the number of input branches for `XOR` operation is more than 2. The exclusion method can be generalized accordingly, and we omit it for space limitation.

Table 2: Illustration of Intermediate Variables for Modular Addition Operation.

Distribution of Intermediate Variables	
$z_{n-1} = \underbrace{x_{n-1}}_{a_{n-1,0}} \oplus \underbrace{y_{n-1}}_{b_{n-1,0}}$	
$z_{n-2} = \underbrace{x_{n-2}}_{a_{n-2,0}} \oplus \underbrace{y_{n-2}}_{b_{n-2,0}} \oplus \underbrace{c_{n-2}}_{g_0}$	$\underbrace{c_{n-2}}_{v_0} = \underbrace{x_{n-1}}_{a_{n-1,1}}\ \underbrace{y_{n-1}}_{b_{n-1,1}}$
$z_{n-3} = \underbrace{x_{n-3}}_{a_{n-3,0}} \oplus \underbrace{y_{n-3}}_{b_{n-3,0}} \oplus \underbrace{c_{n-3}}_{g_1}$	$\underbrace{c_{n-3}}_{w_0} = \overbrace{\underbrace{x_{n-2}}_{a_{n-2,1}}\ \underbrace{y_{n-2}}_{b_{n-2,1}}}^{v_1} \oplus \overbrace{(\overbrace{\underbrace{x_{n-2}}_{a_{n-2,2}} \oplus \underbrace{y_{n-2}}_{b_{n-2,2}}}^{m_0})\ \overbrace{c_{n-2}}^{r_0}}^{q_0}$
$z_{n-4} = \underbrace{x_{n-4}}_{a_{n-4,0}} \oplus \underbrace{y_{n-4}}_{b_{n-4,0}} \oplus \underbrace{c_{n-4}}_{g_1}$	$\underbrace{c_{n-4}}_{w_1} = \overbrace{\underbrace{x_{n-3}}_{a_{n-3,1}}\ \underbrace{y_{n-3}}_{b_{n-3,1}}}^{v_2} \oplus \overbrace{(\overbrace{\underbrace{x_{n-3}}_{a_{n-3,2}} \oplus \underbrace{y_{n-3}}_{a_{n-3,2}}}^{m_1})\ \overbrace{c_{n-3}}^{r_1}}^{q_1}$
...	...
$z_1 = \underbrace{x_1}_{a_{1,0}} \oplus \underbrace{y_1}_{b_{1,0}} \oplus \underbrace{c_1}_{g_{n-3}}$	$\underbrace{c_1}_{w_{n-4}} = \overbrace{\underbrace{x_2}_{a_{2,1}}\ \underbrace{y_2}_{b_{2,1}}}^{v_{n-3}} \oplus \overbrace{(\overbrace{\underbrace{x_2}_{a_{2,2}} \oplus \underbrace{y_2}_{b_{2,2}}}^{m_{n-4}})\ \overbrace{c_2}^{r_{n-4}}}^{q_{n-4}}$
$z_0 = x_0 \oplus y_0 \oplus c_0$	$\underbrace{c_0}_{w_{n-3}} = \overbrace{\underbrace{x_1}_{a_{1,1}}\ \underbrace{y_1}_{b_{1,1}}}^{v_{n-2}} \oplus \overbrace{(\overbrace{\underbrace{x_1}_{a_{1,2}} \oplus \underbrace{y_1}_{b_{1,2}}}^{m_{n-3}})\ \overbrace{c_1}^{r_{n-3}}}^{q_{n-3}}$

Let $\boldsymbol{x} = (x_0, x_1, \ldots, x_{n-1})$, $\boldsymbol{y} = (y_0, y_1, \ldots, y_{n-1})$, and $\boldsymbol{z} = (z_0, z_1, \ldots, z_{n-1})$, which is the modular addition of $\boldsymbol{x}$ and $\boldsymbol{y}$, be n-bit vectors. Then the Boolean function of z_i can be iteratively expressed as follows[8].

$$\begin{aligned} &z_{n-1} = x_{n-1} \oplus y_{n-1} \oplus c_{n-1},\ c_{n-1} = 0, \\ &z_i = x_i \oplus y_i \oplus c_i,\ c_i = x_{i+1} \cdot y_{i+1} \oplus (x_{i+1} \oplus y_{i+1}) \cdot c_{i+1}, \\ &i = n-2, n-3, \ldots, 0. \end{aligned} \tag{1}$$

In this way, the modular addition can be decomposed into **Copy**, `AND`, and `XOR` operations, and the model to depict its propagation is summarized as follows.

Model 4 (Modular Addition) *Let $(a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}, d_0, \ldots, d_{n-1})$ be a division trail of n-bit modular addition operation, to describe the division property propagation, the **Copy**, `AND`, and `XOR` models should be applied in the order*

[8] Note that the bit positions are labeled in big-endian.

specified as follows,

$$
\begin{cases}
(a_{n-1}) \xrightarrow{\textbf{\textit{Copy}}} (a_{n-1,0}, a_{n-1,1}) \\
(b_{n-1}) \xrightarrow{\textbf{\textit{Copy}}} (b_{n-1,0}, b_{n-1,1}) \\
(a_{n-1,0}, b_{n-1,0}) \xrightarrow{XOR} (d_{n-1}) \\
(a_{n-1,1}, b_{n-1,1}) \xrightarrow{AND} (v_0) \\
(v_0) \xrightarrow{\textbf{\textit{Copy}}} (g_0, r_0) \\
(a_{n-2}) \xrightarrow{\textbf{\textit{Copy}}} (a_{n-2,0}, a_{n-2,1}, a_{n-2,2}) \\
(b_{n-2}) \xrightarrow{\textbf{\textit{Copy}}} (b_{n-2,0}, b_{n-2,1}, b_{n-2,2}) \\
\left.\begin{array}{l}
(a_{n-i,0}, b_{n-i,0}, g_{i-2}) \xrightarrow{XOR} (d_{n-i}) \\
(a_{n-i,1}, b_{n-i,1}) \xrightarrow{AND} (v_{i-1}) \\
(a_{n-i,2}, b_{n-i,2}) \xrightarrow{XOR} (m_{i-2}) \\
(m_{i-2}, r_{i-2}) \xrightarrow{AND} (q_{i-2}) \\
(v_{i-1}, q_{i-2}) \xrightarrow{XOR} (w_{i-2}) \\
(w_{i-2}) \xrightarrow{\textbf{\textit{Copy}}} (g_{i-1}, r_{i-1}) \\
(a_{n-i-1}) \xrightarrow{\textbf{\textit{Copy}}} (a_{n-i-1,0}, a_{n-i-1,1}, a_{n-i-1,2}) \\
(b_{n-i-1}) \xrightarrow{\textbf{\textit{Copy}}} (b_{n-i-1,0}, b_{n-i-1,1}, b_{n-i-1,2})
\end{array}\right\} \textit{iterated for } i = 2, \ldots, n-2, \\
(a_{1,0}, b_{1,0}, g_{n-3}) \xrightarrow{XOR} (d_1) \\
(a_{1,1}, b_{1,1}) \xrightarrow{AND} (v_{n-2}) \\
(a_{1,2}, b_{1,2}) \xrightarrow{XOR} (m_{n-3}) \\
(m_{n-3}, r_{n-3}) \xrightarrow{AND} (q_{n-3}) \\
(v_{n-2}, q_{n-3}) \xrightarrow{XOR} (w_{n-3}) \\
(a_0, b_0, w_{n-3}) \xrightarrow{XOR} (d_0)
\end{cases}
$$

where $a_{i,j}$, $b_{i,j}$, v_i, m_i, g_i, r_i, q_i, and w_i are intermediate variables, and their usage is illustrated in Table 2. In this model, $(12n-19)$ intermediate variables are introduced in total, which include $(3n-4)$ $a_{i,j}$'s, $(3n-4)$ $b_{i,j}$'s, $(n-1)$ v_i's, $(n-2)$ m_i's, $(n-2)$ g_i's, $(n-2)$ r_i's, $(n-2)$ q_i's, and $(n-2)$ w_i's.

Model 4 deals with the case where the two input branches of the modular addition operation are variables. When it comes to the modular addition of a variable and an unknown constant (subkey), the corresponding propagation models can be deduced similarly as discussed in [36], and we omit it due to space limitation.

To sum up, the bit-based division property propagations through all kinds of basic operations in ARX ciphers are converted into sets of logical equations. We first construct SAT model which characterizes one round bit-based division property propagation, then an SAT problem depicting r-round division trails can be obtained by repeating this procedure for r times.

3.2 Initial Division Property and Stopping Rule

We propose a 'dynamic' searching, which can set the initial division property and stopping rule more efficiently. In the C++ interface of CryptoMiniSat, there is a function called `solver()`, which takes 'assumptions' as parameter, so that we can adjust the 'assumptions', instead of the original model, and invoke `solver()` calls to search for integral distinguishers under different initial division properties and output division properties automatically. In our model, 'assumptions' are composed of two parts of logical equations: one is determined by the initial division property, and another is deduced from the stopping rule.

Initial Division Property. Denote $(a_0, a_1, \ldots, a_{n-1})$ the variables representing bit-based division property of the input multi-set. For example, suppose that the initial division property is $\boldsymbol{k}_0 = (0, \underbrace{1, \ldots, 1}_{n-1})$. To evaluate the propagation under $\boldsymbol{k}_0$, we set the first part of the assumptions by logical equations, i.e., $a_0 = 0$, $a_1 = 1$, …, $a_{n-1} = 1$. If we want to test division property under another initial division property, only logical equations involved in the assumptions need to be changed.

Stopping Rule. The stopping rule is formulated according to Proposition 1. When it comes to the bit-based division property, a multi-set $\mathbb{X}$, whose elements take values from $\mathbb{F}_2^n$, does not have integral property if and only if its division property contains all the n unit vectors. Hence, we need to check all the n unit vectors one by one. Denote $(b_0, b_1, \ldots, b_{n-1})$ the variables representing bit-based division property of the output multi-set after r rounds. For each $i \in \{0, 1, \ldots, n-1\}$, we set the second part of the assumptions by $b_i = 1$ and $b_j = 0$ $(j \neq i)$. Together with the initial division property, the two parts of parameters are determined for the `solver()` function, and the searching algorithm can be transformed into an SAT problem. If it is 'satisfiable' for the i-th unit vector, it means that the output division property contains the i-th unit vector. Once it is satisfiable for each unit vector, the output division property contains all unit vectors, and the corresponding multi-set, i.e., the outputs of the r-th round, does not have any integral property, and the propagation should stop and an $(r-1)$-round distinguisher is obtained. Only if there is at least one index j, such that the problem is not satisfiable for the j-th unit vector, we proceed to the $(r+1)$-th round and evaluate the division property in a similar way.

3.3 Algorithms to Find Optimal Distinguishers

According to the discussion of the above subsections, the propagation of division property for ARX cipher is depicted by a system of logical equations in CNF. Some logical formulas can be dynamically adjusted according to different initial division properties of the input set and final division properties of the output set, while the others corresponding to r-round propagations remain the same.

To obtain an optimal integral distinguisher, many candidates of initial division properties need to be tested. However, we could not afford such computations for too many candidates in practice.

In order to break through the difficulty, we put forward an efficient searching approach, which is composed of two algorithms. The first one restricts the search scope of initial division property and detects the number of rounds of the optimal distinguisher achieved under our model. For the instance of SHACAL-2, the search scope is significantly reduced from 256 bits to 17 bits. The second one detects the concrete optimal distinguishers efficiently based on the first algorithm's output. With these two algorithms, we drastically reduce the number of initial division properties required to be evaluated. For example, for the 17-round distinguisher with data complexity 2^{241} chosen plaintexts for SHACAL-2, which is provided in Section 5.1, the direct search requires us to test $\sum_{i=1}^{256-241}\binom{256}{i} \approx 2^{79.24}$ initial division properties. While in our algorithms, only 410 initial division properties are tested, and the distinguisher is identified.

The design of the two algorithms is based on the *embedded property* below. For different initial division properties $\boldsymbol{k}_0$ and $\boldsymbol{k}_1$ *s.t.*, $\boldsymbol{k}_0 \succeq \boldsymbol{k}_1$, there in no need to test $\boldsymbol{k}_1$, if the output multi-set under $\boldsymbol{k}_0$ does not have integral property, likewise, it is not necessary to test $\boldsymbol{k}_0$, if the output multi-set under $\boldsymbol{k}_1$ has integral property.

Proposition 2 (Embedded Property). *Let E_r be an r-round iterated encryption algorithm, f be the round function, which only composes of* ***Substitution****,* ***Copy****,* `XOR`*,* ***Split****, and* ***Concatenation*** *operations. Suppose that the input and the output take values from $\mathbb{F}_2^n = \mathbb{F}_2^{\ell_0} \times \mathbb{F}_2^{\ell_1} \times \cdots \times \mathbb{F}_2^{\ell_{m-1}}$, $\boldsymbol{k}_0$ and $\boldsymbol{k}_1$ are two initial division properties with $W(\boldsymbol{k}_0) \succeq W(\boldsymbol{k}_1)$. If the output multi-set under $\boldsymbol{k}_0$ does not have integral property, then the output multi-set under $\boldsymbol{k}_1$ has no integral property.*

Proof: Define

$$\mathbb{S}_{\boldsymbol{k}}^n = \{\boldsymbol{a} = (a_0, a_1, \ldots, a_{m-1}) | W(\boldsymbol{a}) \succeq W(\boldsymbol{k})\},$$

and

$$\mathbb{S}_{\mathbb{K}}^n = \bigcup_{\boldsymbol{k} \in \mathbb{K}} \mathbb{S}_{\boldsymbol{k}}^n.$$

Suppose that there are two sets $\mathbb{K}_0$ and $\mathbb{K}_1$ belonging to $\mathbb{Z}_{\ell_0} \times \mathbb{Z}_{\ell_1} \times \cdots \times \mathbb{Z}_{\ell_{m-1}}$, with $\mathbb{S}_{\mathbb{K}_0}^n \subseteq \mathbb{S}_{\mathbb{K}_1}^n$. $\mathcal{D}_{\mathbb{K}_0}^n \xrightarrow{f} \mathcal{D}_{\mathbb{K}_0'}^n$ and $\mathcal{D}_{\mathbb{K}_1}^n \xrightarrow{f} \mathcal{D}_{\mathbb{K}_1'}^n$ stand for the division property propagations through one round. By the definition of division property, it is sufficient to prove that $\mathbb{S}_{\mathbb{K}_0'}^n \subseteq \mathbb{S}_{\mathbb{K}_1'}^n$, which can be accomplished by separately proving for every basic operation. We take the substitution operation as an example, and the other operations can be proved similarly.

Now, denote $\mathcal{D}_{\mathbb{K}_0}^n \xrightarrow{\mathrm{S}} \mathcal{D}_{\mathbb{K}_0'}^n$ and $\mathcal{D}_{\mathbb{K}_1}^n \xrightarrow{\mathrm{S}} \mathcal{D}_{\mathbb{K}_1'}^n$ the division property propagations through substitution layer, where $\mathbb{S}_{\mathbb{K}_0}^n \subseteq \mathbb{S}_{\mathbb{K}_1}^n$. For every $\boldsymbol{k}_0' \in \mathbb{K}_0'$, there exists $\boldsymbol{k}_0 \in \mathbb{K}_0$, such that $(\boldsymbol{k}_0, \boldsymbol{k}_0')$ constitutes a division trail of the substitution

operation. Since $\mathbb{S}^n_{\mathbb{K}_0} \subseteq \mathbb{S}^n_{\mathbb{K}_1}$, there will be a $\boldsymbol{k}_1 \in \mathbb{K}_1$ with $W(\boldsymbol{k}_0) \succeq W(\boldsymbol{k}_1)$. By Rule 1, we have $W(\boldsymbol{k}'_0) \succeq W(\boldsymbol{k}'_1)$, which implies that $\mathbb{S}^n_{\boldsymbol{k}'_0} \subseteq \mathbb{S}^n_{\boldsymbol{k}'_1}$. Thus,

$$\mathbb{S}^n_{\mathbb{K}'_0} = \bigcup_{\boldsymbol{k}'_0 \in \mathbb{K}'_0} \mathbb{S}^n_{\boldsymbol{k}'_0} \subseteq \bigcup_{\boldsymbol{k}'_1 \in \mathbb{K}'_1} \mathbb{S}^n_{\boldsymbol{k}'_1} = \mathbb{S}^n_{\mathbb{K}'_1}.$$

□

Algorithm 1: Detecting the Maximum Number of Rounds & Restricting the Search Scope. Denote the n vectors with Hamming weight $n-1$ as $\boldsymbol{in}_i = (\underbrace{1,\ldots,1}_{i},0,\underbrace{1,\ldots,1}_{n-i-1})$, $0 \leqslant i \leqslant n-1$. Let $\boldsymbol{out}_j = (\underbrace{0,\ldots,0}_{j},1,\underbrace{0,\ldots,0}_{n-j-1})$, $0 \leqslant j \leqslant n-1$, be the n unit vectors. For $0 \leqslant i \leqslant n-1$, we evaluate the bit-based division property propagation under the initial division property $\boldsymbol{in}_i$, and check whether the output division property of the r-th round contains all n unit vectors, i.e., the problem is satisfiable for each $\boldsymbol{out}_j$ $(0 \leqslant j \leqslant n-1)$ under the fixed $\boldsymbol{in}_i$. If for all $\boldsymbol{in}_i$ $(0 \leqslant i \leqslant n-1)$ and $\boldsymbol{out}_j$ $(0 \leqslant j \leqslant n-1)$, the problem is satisfiable, we conclude that $(r-1)$ is the maximum number of rounds based on our model. Otherwise, we proceed to the $(r+1)$-th round and evaluate the division property in a similar way. When the maximum number of rounds r_m is determined, the index i of the corresponding $\boldsymbol{in}_i$ leading to the longest distinguisher is stored in a set $\mathbb{S}$. The output of Algorithm 1 is the maximum number of round r_m and an index set $\mathbb{S}$.

Although we have detected r_m-round distinguishers, the data requirement to implement the integral cryptanalysis is 2^{n-1}. And the distinguisher with lower data complexity is more interesting, so we proceed Algorithm 2 to optimize the distinguishers obtained in Algorithm 1.

Algorithm 2: Detecting the Optimal Distinguisher. Let the index set $\mathbb{S} = \{j_0, j_1, \ldots, j_{|\mathbb{S}|-1}\}$ be the output of Algorithm 1. With Proposition 2, we claim that the elements in the complementary set $\overline{\mathbb{S}} = \{0,1,\ldots,n-1\}\backslash\mathbb{S}$ of $\mathbb{S}$ refer to the 'necessary' bit indexes to obtain an r_m-round integral distinguisher. In other words, if any bit whose index belongs to $\overline{\mathbb{S}}$ is set to '0' in the initial division property, the division property after r_m-round propagation will have no integral property. In this sense, we call $\overline{\mathbb{S}}$ the *necessary set*, whose elements are called *necessary indexes*, and the corresponding bit must be fixed to '1', while, $\mathbb{S}$ is called the *sufficient set*, and the elements in $\mathbb{S}$ are called *sufficient indexes*.

To reduce the data complexity, we need to analyze whether the bits with sufficient indexes can be set to '0'. The possibility of reducing data complexity lies in the size of $\mathbb{S}$. If $|\mathbb{S}| = 1$, there is no margin to further reduce the data complexity, and we obtain integral distinguishers with data complexity 2^{n-1} chosen plaintexts. In case of $|\mathbb{S}| > 1$, we firstly set all bits corresponding to $\overline{\mathbb{S}}$ in initial division property to '1' while the other bits are set to '0', and check whether there is zero-sum bit after r_m-round propagation. If it is indeed the case, we get an integral distinguisher with data complexity $2^{n-|\mathbb{S}|}$ chosen plaintexts.

Algorithm 1: Detecting the Maximum Number of Rounds & Restricting the Search Scope

Input: Objective algorithm E
Output: The maximum number of rounds r_m of integral distinguisher, the index set $\mathbb{S}$

```
1  r = 0, 𝕊 = ∅, flag = 1, r_m = 0;
2  while flag==1 do
3      r = r + 1;
4      flag = 0;
5      for i = 0; i < n do
6          let the initial division property be in_i;
7          for j = 0; j < n do
8              let the output division property be out_j;
9              solve the r-round SAT problem under the assumptions;
10             if the problem is not satisfiable then
11                 flag=1;
12                 break;
13         if flag == 1 then
14             break;
15 r = r − 1, r_m = r;
16 if r_m == 0 then
17     return r_m, 𝕊;
18 for i = 0; i < n do
19     let the initial division property be in_i;
20     evaluate its division property after r_m-round propagation;
21     if there is zero-sum bit then
22         𝕊 = 𝕊 ∪ {i};
23         continue;
24 return r_m, 𝕊;
```

Otherwise, we gradually increase the number of '1's in the positions indicated by the sufficient indexes, and check whether zero-sum bit exists or not. The concrete description of this procedure can be found in Algorithm 2. After executing this algorithm, the return value will be the optimal distinguishers under our model.

Remark 2. Note that Step 8 in Algorithm 2 requests us to check out $\frac{|\mathbb{S}|!}{(|\mathbb{S}|-t)!\cdot t!}$ different initial division properties. When $|\mathbb{S}|$ is very large, the time taken to perform this **for** loop gradually increases with t growing. But, for all the ciphers analyzed in this paper, $|\mathbb{S}|$ is not very large and the runtime is acceptable.

4 Automatic Search of Word-Based Division Property

When the state of the cipher is very large, such as 256-bit, and the involved operations are very complicated, it is hard to trace the division property prop-

agation at the bit level. In this section, we concentrate on automatic search of word-based division property efficiently. First, we study how to model division property propagations of basic operations by logical formulas at the word level. Secondly, by exclusion method, we construct formulas to depict the possible propagations calculated by Substitution rule. By setting initial division property and stopping rule rationally, the problem of searching division property can be transformed into an SMT problem, which is a generalization of SAT and can be efficiently settled with some openly available solvers.

4.1 Models of Basic Operations at the Word Level

We study the division property propagations of the basic operations at the word level. Different from Section 3, the input and output are variables in $\mathbb{F}_2^n$, and more kinds of formulas, such as inequalities, can be handled by SMT, so that the translation from the rules introduced in Section 2.2 to constraints are more flexible. We just list the models as follows.

Model 5 (Word-based Copy) *Denote* $(a) \xrightarrow{\textit{Copy}} (b_0, b_1)$ *a division trail of an* n*-bit* ***Copy*** *function, the following constraints are sufficient to describe the division property propagation of* ***Copy*** *operation,*

$$\begin{cases} a \leqslant n \\ b_0 \leqslant n \\ b_1 \leqslant n \\ a = b_0 + b_1 \end{cases}.$$

Model 6 (Word-based XOR) *Denote* $(a_0, a_1) \xrightarrow{\texttt{XOR}} (b)$ *a division trail of* n*-bit* `XOR` *operation, the following constraints are sufficient to depict the division property propagation of* `XOR` *operation,*

$$\begin{cases} a_0 \leqslant n \\ a_1 \leqslant n \\ b \leqslant n \\ a_0 + a_1 = b \end{cases}.$$

Model 7 (Split) *Let* F *be the split function in Rule 4. Denote* $(a) \xrightarrow{F} (b_0, b_1)$ *a division trail of* F*, the following constraints are sufficient to describe the division property propagation of* ***Split*** *operation,*

$$\begin{cases} a \leqslant n \\ b_0 \leqslant n_0 \\ b_1 \leqslant n - n_0 \\ a = b_0 + b_1 \end{cases}.$$

Model 8 (Concatenation) *Let* F *be the concatenation function in Rule 5. Denote* $(a_0, a_1) \xrightarrow{F} (b)$ *a division trail of* F*, the following constraints are sufficient*

Algorithm 2: Detecting the Optimal Distinguisher

Input: Objective algorithm E, the maximum number of rounds r_m, the sufficient set $\mathbb{S}$

Output: A list *List* representing Optimal integral distinguishers

$flag = 0$, $List = \emptyset$, $\boldsymbol{k_0} = (0, 0, \ldots, 0)$;
$k_0 = 0$, $k_1 = 0$, ..., $k_{n-1} = 0$;
for $i = 0; i < n$ **do**
    if $i \notin \mathbb{S}$ **then**
        $k_i = 1$;
$t = 0$;
while $flag == 0$ **do**
    for *every t-tuple* $(i_0, i_1, \ldots, i_{t-1})$ *of* $\mathbb{S}$ **do**
        for $i \in \mathbb{S}$ **do**
            if $i \in \{i_0, i_1, \ldots, i_{t-1}\}$ **then**
                $k_i = 1$;
            else if $i \in \mathbb{S} \backslash \{i_0, i_1, \ldots, i_{t-1}\}$ **then**
                $k_i = 0$;
        let the initial division property be $\boldsymbol{k}_0 = (k_0, k_1, \ldots, k_{n-1})$;
        evaluate its bit-based division property after r_m-round propagation;
        if *there is zero-sum bit* **then**
            $flag = 1$;
            break;
    $t = t + 1$;
$t = t - 1$;
for *every t-tuple* $(i_0, i_1, \ldots, i_{t-1})$ *of* $\mathbb{S}$ **do**
    $InActive = \mathbb{S} \backslash \{i_0, i_1, \ldots, i_{t-1}\}$;
    for $i \in \mathbb{S}$ **do**
        if $i \in \{i_0, i_1, \ldots, i_{t-1}\}$ **then**
            $k_i = 1$;
        else if $i \in \mathbb{S} \backslash \{i_0, i_1, \ldots, i_{t-1}\}$ **then**
            $k_i = 0$;
    let the initial division property be $\boldsymbol{k}_0 = (k_0, k_1, \ldots, k_{n-1})$;
    evaluate its bit-based division property after r_m-round propagation;
    $ZeroSum = \emptyset$;
    for $i = 0; i < n$ **do**
        if *the i-th output bit satisfies zero-sum property* **then**
            $ZeroSum = ZeroSum \cup \{i\}$;
    if $ZeroSum \neq \emptyset$ **then**
        $List = List \cup \{InActive, ZeroSum\}$;
return *List*;

to depict the division property propagation of ***Concatenation*** *operation,*

$$\begin{cases} a_0 \leqslant n_0 \\ a_1 \leqslant n_1 \\ b \leqslant n_0 + n_1 \\ a_0 + a_1 = b \end{cases}.$$

Many ciphers take Maximum Distance Separable (MDS) matrices over finite field as linear mappings, such as the `MixColumn` operation for AES [28]. Todo [41] proposed a dedicated function called `Partition` to handle the division property propagation through `MixColumn` operation. We generalize it into SMT model in order to deal with some ciphers involving MDS matrices.

Model 9 (`Partition`/`MixColumn`) *Let $F(x) = M \cdot x$, where M is an MDS matrix over $(\mathbb{F}_2^m)^s$. Denote $(a_0, a_1, \ldots, a_{s-1}) \xrightarrow{F} (b_0, b_1, \ldots, b_{s-1})$ a division trail, the following constraints are sufficient to propagate the division property,*

$$\begin{cases} a_i \leqslant m, i = 0, 1, \ldots, s-1 \\ b_j \leqslant m, j = 0, 1, \ldots, s-1 \\ a_0 + a_1 + \cdots + a_{s-1} = b_0 + b_1 + \cdots + b_{s-1} \end{cases}.$$

4.2 Modelling S-box

Since conventional division property is propagated at the word level, we do not need to precisely depict S-box, and use Rule 1 instead. By Rule 1, we find that the output multi-set follows $\mathcal{D}^m_{\lceil \frac{k}{d} \rceil}$ if the input multi-set satisfies $\mathcal{D}^m_k$ for an m-bit S-box with degree d. Accordingly, we deduce possible propagations for S-box, which are converted into SMT model by exclusion method mentioned in Section 3.

Model 10 (4-bit S-box with Degree 3) *Denote $(x) \xrightarrow{S_{(4)}} (y)$ a division trail of 4-bit S-box $S_{(4)}$, whose algebraic degree is 3, where $x = (x[0], x[1], x[2])$ and $y = (y[0], y[1], y[2])$ are supposed to be 3-bit vectors. Then, the following constraints are sufficient to describe the propagation of division property,*

$$\begin{cases} x \leqslant 4 \\ y \leqslant 4 \\ x[0] \vee \overline{y[0]} = 1 \\ \overline{x[0]} \vee x[1] \vee x[2] \vee y[0] = 1 \\ \overline{y[1]} = 1 \\ x[0] \vee \overline{x[1]} \vee y[0] \vee y[1] \vee y[2] = 1 \\ x[0] \vee x[1] \vee \overline{x[2]} \vee y[0] \vee y[1] \vee y[2] = 1 \\ x[0] \vee x[1] \vee x[2] \vee y[0] \vee y[1] \vee \overline{y[2]} = 1 \end{cases}.$$

Proof: Note that for a 4-bit S-box with algebraic degree 3, the possible propagations are $(0) \xrightarrow{S_{(4)}} (0)$, $(1) \xrightarrow{S_{(4)}} (1)$, $(2) \xrightarrow{S_{(4)}} (1)$, $(3) \xrightarrow{S_{(4)}} (1)$, and $(4) \xrightarrow{S_{(4)}} (4)$,

and the natural constraints deduced from Rule 1 are $x \leqslant 4$ and $y \leqslant 4$. After adding these two natural constraints, the number of possible combinations of $(x[0], x[1], x[2], y[0], y[1], y[2])$ reduces to 25, which are

$$\begin{array}{l}
\textbf{(0,0,0,0,0,0)}, (0,0,0,0,0,1), (0,0,0,0,1,0), (0,0,0,0,1,1), (0,0,0,1,0,0), \\
(0,0,1,0,0,0), \textbf{(0,0,1,0,0,1)}, (0,0,1,0,1,0), (0,0,1,0,1,1), (0,0,1,1,0,0), \\
(0,1,0,0,0,0), \textbf{(0,1,0,0,0,1)}, (0,1,0,0,1,0), (0,1,0,0,1,1), (0,1,0,1,0,0), \\
(0,1,1,0,0,0), \textbf{(0,1,1,0,0,1)}, (0,1,1,0,1,0), (0,1,1,0,1,1), (0,1,1,1,0,0), \\
(1,0,0,0,0,0), (1,0,0,0,0,1), (1,0,0,0,1,0), (1,0,0,0,1,1), \textbf{(1,0,0,1,0,0)}.
\end{array}$$

The five vectors in bold are what we expect. After observation, $(0, *, *, 1, *, *)$, $(1, 0, 0, 0, *, *)$, $(*, *, *, *, 1, *)$, $(0, 1, *, 0, 0, 0)$, $(0, 0, 1, 0, 0, 0)$ and $(0, 0, 0, 0, 0, 1)$ are impossible cases, where $*$ takes 0 or 1.

In order to eliminate $(0, *, *, 1, *, *)$, we assert $x[0] \vee \overline{y[0]} = 1$. With this assertion, $(x[0], x[1], x[2], y[0], y[1], y[2])$ cannot take values of the form $(0, *, *, 1, *, *)$. After eliminating all impossible cases one by one, we obtain the set of logical formulas to describe division property propagation of $S_{(4)}$ □

For 8-bit S-box with degree 7, possible propagations are $(0) \to (0)$, $(1) \to (1)$, $(2) \to (1)$, $(3) \to (1)$, $(4) \to (1)$, $(5) \to (1)$, $(6) \to (1)$, $(7) \to (1)$, and $(8) \to (8)$, and the model can be constructed in a similar way.

Model 11 (8-bit S-box with Degree 7) *Denote* $(x) \xrightarrow{S_{(8)}} (y)$ *a division trail of 8-bit S-box* $S_{(8)}$*, whose algebraic degree is 7, where* $x = (x[0], x[1], x[2], x[3])$ *and* $y = (y[0], y[1], y[2], y[3])$ *are supposed to be 4-bit vectors. Then, the following constraints are sufficient to describe the possible propagations,*

$$\begin{cases}
x \leqslant 8 \\
y \leqslant 8 \\
\overline{x[0]} \vee y[0] = 1 \\
x[0] \vee \overline{y[0]} = 1 \\
y[1] = 0 \\
\overline{y[2]} = 0 \\
\overline{x[3]} \vee y[0] \vee y[1] \vee y[2] \vee y[3] = 1 \\
\overline{x[2]} \vee y[0] \vee y[1] \vee y[2] \vee y[3] = 1 \\
\overline{x[1]} \vee y[0] \vee y[1] \vee y[2] \vee y[3] = 1 \\
x[0] \vee x[1] \vee x[2] \vee x[3] \vee y[0] \vee y[1] \vee y[2] \vee \overline{y[3]} = 1
\end{cases}.$$

For other types of S-boxes, exclusion method can be applied and constraints to depict division property propagations can be constructed similarly.

4.3 Initial Division Property and Stopping Rule

Just as in Section 3, to make the searching algorithm dynamic, the initial division property and stopping rule are inserted into assumptions. In the Python interface of STP, the function, which accepts 'assumptions' as parameter, is called `check()`.

Denote $(a_0, a_1, \ldots, a_{m-1})$ the variables representing division property of the input multi-set. For example, suppose that the initial division property is $\boldsymbol{k} = (k_0, k_1, \ldots, k_{m-1})$. To propagate division property under $\boldsymbol{k}$, we set the first part of the assumptions by logical formulas, i.e., $a_0 = k_0$, $a_1 = k_1$, ..., and $a_{m-1} = k_{m-1}$. Only logical formulas involved in the assumptions are required to be replaced if we want to test division property under another initial division property.

Restricted to conventional division property, Proposition 1 claims that a multi-set $\mathbb{X} \in \mathbb{F}_2^n = \mathbb{F}_2^{\ell_0} \times \mathbb{F}_2^{\ell_1} \times \cdots \times \mathbb{F}_2^{\ell_{m-1}}$ does not have integral property if and only if its division property contains all vectors with vectorial Hamming weight being 1. In order to determine whether r-round integral property exists or not under a fixed initial division property, we make m `check()` calls to test m vectors with vectorial Hamming weight 1. If all the corresponding SMT problems are satisfiable, the r-round output set has no integral property and an $(r-1)$-round distinguisher is obtained. Otherwise, we go on to the $(r+1)$-th round and evaluate the division property in a similar way.

5 Applications

In this section, we provide some new distinguishers based on the searching methods proposed in Section 3 and 4. We first present results for some ARX ciphers, whose integral distinguishers are obtained by evaluating bit-based division property, and then turn to the word-based division property of some specific ciphers.

5.1 Bit-Based Division Properties for ARX Ciphers

Application to SHACAL-2. SHACAL-2 [13] is a 256-bit block cipher and has been selected as one of the four block ciphers by NESSIE. Its round function is based on the compression function of the hash function SHA-2 [27], and is iterated for 64 times. SHACAL-2 supports variable key lengths up to 512 bits, yet it should not be used with a key shorter than 128 bits. An illustration of the round function can be found in Fig. 1, where K^r and W^r are round key and round constant, Maj, Ch, $\sum_0$, and $\sum_1$ are defined as follows,

$$\begin{aligned} Maj(X,Y,Z) &= (X \cdot Y) \oplus (X \cdot Z) \oplus (Y \cdot Z), \\ Ch(X,Y,Z) &= (X \cdot Y) \oplus (\overline{X} \cdot Z), \\ \sum\nolimits_0(X) &= (X \ggg 2) \oplus (X \ggg 13) \oplus (X \ggg 22), \\ \sum\nolimits_1(X) &= (X \ggg 6) \oplus (X \ggg 11) \oplus (X \ggg 25). \end{aligned}$$

Since the values of K^r and W^r do not influence the bit-based division property propagation, and we will not introduce them here. For more information, please refer to [13].

Firstly, Algorithm 1 in Section 3.3 is implemented and we find that the longest distinguisher under our model can achieve 17 rounds. At the same time,

we obtain the sufficient set $\mathbb{S} = \{22-31, 153-159\}$. Then, for $r = 17$ and $\mathbb{S}$, Algorithm 2 is performed. Finally, we obtain a 17-round integral distinguisher with data complexity 2^{241} chosen plaintexts, which is

$$\text{Inactive Bits: } \{23-31, 154-159\} \xrightarrow{17 \text{ Rounds}} \text{Zero-sum Bits: } \{249-255\},$$

where the bit indexes for the input and output are labeled as 0, 1, ..., 255 from left to right, and the bit indexes are labeled in a similar way in the remaining of this subsection. In order to identify this distinguisher, we try 256 initial division properties when implementing Algorithm 1, and $1+\binom{17}{1}+\binom{17}{2} = 154$ initial division properties are evaluated when performing Algorithm 2. In total, with 410 tests under different initial division properties, we obtain the optimal distinguisher, while $\sum_{i=1}^{256-241}\binom{256}{i} \approx 2^{79.24}$ initial division properties are required to be tested for the direct search instead of using Algorithm 1 and Algorithm 2.

As far as we know, the best integral distinguisher in the literature is the 13-round one proposed in [30], and the newly obtained one covers four more rounds.

Applications to Other ARX Ciphers. Besides SHACAL-2, many ARX ciphers are analyzed, including LEA [14], HIGHT [15], and SPECK family of block ciphers [3], and we only list the results for space limitation.

For LEA, we obtain an 8-round integral distinguisher with data complexity 2^{118} chosen plaintexts, which is

$$\text{Inactive Bits: } \{27-31, 59-63\} \xrightarrow{8 \text{ Rounds}} \text{Zero-sum Bits: } \{36\}.$$

Comparing to the 7-round distinguishers based on MILP method provided in [36], we gain one more round.

Six integral distinguishers with data complexity 2^{63} chosen plaintexts are detected for HIGHT, which are

$$\begin{aligned}
&\text{Inactive Bits: } \{14\} \xrightarrow{18 \text{ Rounds}} \text{Zero-sum Bits: } \{6,7\},\\
&\text{Inactive Bits: } \{15\} \xrightarrow{18 \text{ Rounds}} \text{Zero-sum Bits: } \{6,7\},\\
&\text{Inactive Bits: } \{31\} \xrightarrow{18 \text{ Rounds}} \text{Zero-sum Bits: } \{7\},\\
&\text{Inactive Bits: } \{46\} \xrightarrow{18 \text{ Rounds}} \text{Zero-sum Bits: } \{38,39\},\\
&\text{Inactive Bits: } \{47\} \xrightarrow{18 \text{ Rounds}} \text{Zero-sum Bits: } \{38,39\},\\
&\text{Inactive Bits: } \{63\} \xrightarrow{18 \text{ Rounds}} \text{Zero-sum Bits: } \{39\}.
\end{aligned}$$

Note that the third one and the last one are same to the 18-round distinguishers in [36], which are obtained under MILP method. And the other four distinguishers we identified have more zero-sum bits under the same data requirement.

For all versions of SPECK family of block ciphers, we obtain 6-round integral distinguishers. The data requirements are 2^{31} for SPECK32, 2^{45} for SPECK48, 2^{61} for SPECK64, 2^{93} for SPECK96, and 2^{125} for SPECK128.

All of the experiments are conducted on a server, and we use at most four 2.30GHz Intel® Xeon® CPU E5-2670 v3 processors. All the SAT based experiments are implemented by the C++ interface of CryptoMiniSat5, using at most 4 threads. The runtimes to obtain the optimal distinguishers for SHACAL-2, LEA, and HIGHT are 6 hours, 30 minutes, and 15 minutes, respectively, and the runtimes for all variants of SPECK take less than 6 minutes.

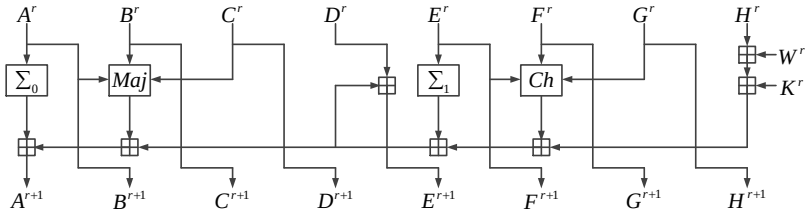

Fig. 1: The Round Function of SHACAL-2.

5.2 Word-Based Division Property for Some Specific Ciphers

Application to CLEFIA. CLEFIA [31] is a 128-bit block cipher supporting key lengths of 128, 192, and 256 bits, and it has been adopted as one of the ISO/IEC international standards in lightweight cryptography. The number of rounds, are 18, 22 and 26 for 128-bit, 192-bit and 256-bit keys, respectively. The round function follows a 4-branch Type-2 Generalized Feistel Network [48] with two parallel F functions (F_0, F_1). The 128-bit state value can be regarded as concatenation of four 32-bit words, and the input of the r-th round is denoted by $(X^r[0], X^r[1], X^r[2], X^r[3])$. One round of encryption is illustrated in Fig. 2, where $RK^r[0]$ and $RK^r[1]$ denote round keys.

Aiming at searching integral distinguishers for CLEFIA as long as possible, we first evaluate the division property under 16 initial division properties $\boldsymbol{in}_i$, $0 \leqslant i \leqslant 15$, whose i-th element is set to 7, and the others are set to 8. Then, we obtain eight 10-round integral distinguishers with data complexity 2^{127} chosen plaintexts. We also evaluate the division property under another 16 initial division properties $\boldsymbol{in}'_i$, $0 \leqslant i \leqslant 15$, whose i-th element is set to 6, and the others are set to 8. However, there is no integral property after 10-round propagation under $\boldsymbol{in}'_i$. Besides, 120 initial division properties with two elements being 7 and the others being 8 are also considered, and no integral property is detected. Thus, the 10-round integral distinguishers with data complexity 2^{127} chosen plaintexts probably are the best integral distinguishers using word-based division property. The initial division properties of these 10-round distinguishers are listed as follows.

$$(7,8,8,8,\ 8,8,8,8,\ 8,8,8,8,\ 8,8,8,8),\ (8,7,8,8,\ 8,8,8,8,\ 8,8,8,8,\ 8,8,8,8),$$

$(8,8,7,8,\ 8,8,8,8,\ 8,8,8,8,\ 8,8,8,8)$, $(8,8,8,7,\ 8,8,8,8,\ 8,8,8,8,\ 8,8,8,8)$,
$(8,8,8,8,\ 8,8,8,8,\ 7,8,8,8,\ 8,8,8,8)$, $(8,8,8,8,\ 8,8,8,8,\ 8,7,8,8,\ 8,8,8,8)$,
$(8,8,8,8,\ 8,8,8,8,\ 8,8,7,8,\ 8,8,8,8)$, $(8,8,8,8,\ 8,8,8,8,\ 8,8,8,7,\ 8,8,8,8)$.

After 10-round propagation, all the 10-round distinguishers have eight zero-sum bytes, which are labeled as $\{4-7, 12-15\}$, and the bytes are labeled as 0, 1, ..., 15 from left to right.

To our knowledge, the longest integral distinguishers for CLEFIA cover 9 rounds [19, 29], and these newly found distinguishers achieve one more round. With the 10-round distinguishers, we can recover the key of 13-round CLEFIA-128 with one more round than [19], where the precomputation, partial sum technique and exhaustive search can be adopted similarly. The data, time and memory complexities are 2^{127} chosen plaintexts, 2^{120} encryptions and 2^{100} bytes, respectively. The integral attacks for CLEFIA-192 and CLEFIA-256 can be improved by one round, too.

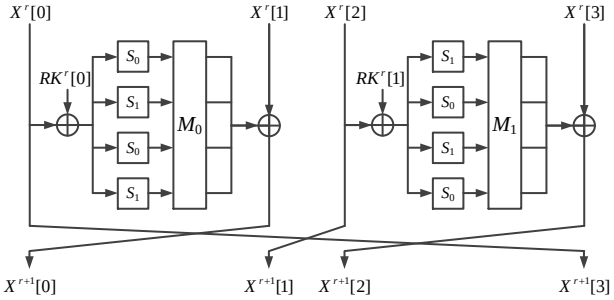

Fig. 2: Round Function of CLEFIA.

Applications to Other Ciphers. We also implement the method in Section 4 to search integral distinguishers for many other ciphers.

For the internal block cipher of Whirlpool [1], comparing to the results given by Todo [41], we improve the data complexities of integral distinguishers for different rounds, which can be found in Table 3. For Rijndael-192 and Rijndael-256 [8], we extend the length of distinguishers comparing to the best results proposed by Todo [41], and the experimental results can be found in Table 3. The integral distinguishers for Whirlpool, Rijndael-192, and Rijndael-256 are provided in Appendix A.

We also implement our automatic tool to search integral distinguishers for MISTY1, MISTY2 [21], and KASUMI [33]. For MISTY1, we obtain the same distinguisher found by Todo [40]. As to MISTY2, a 7-round integral distinguisher with data complexity 2^{32} chosen plaintexts is found, which is same to the best one proposed in [34]. A 5-round integral distinguisher starting from the second round with data complexity 2^{48} chosen plaintexts is obtained for KASUMI. Comparing to the best 5-round one proposed in [35] with data complexity 2^{53} chosen plaintexts by using division property, our newly found distinguisher requires less data.

Table 3: Data Requirements to Construct r-Round Integral Distinguishers.

Cipher	$\log_2(Data)$				Reference
	$r=3$	$r=4$	$r=5$	$r=6$	
Rijndael-192	8	24	64	160	Section 5.2
	56	176	-	-	[41]
Rijndael-256	8	16	32	160	Section 5.2
	56	232	-	-	[41]
Whirlpool	8	56	384	-	Section 5.2
	56	344	488	-	[41]

All the SMT based tests are implemented in the Python interface of STP2.0, using single thread. The runtimes for all the ciphers analyzed in this section only take few minutes.

6 Conclusion

In this paper, we propose the automatic searching tools for the integral distinguishers based on bit-based division property for ARX ciphers and word-based division property. For ARX ciphers, the automatic searching tool relying on SAT instead of MILP is constructed, since SAT method is more suitable in the search of ARX ciphers' differential/linear characteristics. First, the models, which are composed of logical formulas in CNF, to describe bit-based division property propagations for three basic operations, i.e., **Copy**, `AND`, and `XOR`, are provided by exclusion method. Then, we give the model of the modular addition based on the three basic models. After setting initial division property and stopping rule appropriately, the problem of searching integral distinguishers using bit-based division property for ARX ciphers can be converted into an SAT problem. Besides, to get the optimal distinguisher, two algorithms are proposed. The first one restricts the search scope of initial division property and detects the round of optimal distinguisher achieved under our model. The second one detects the concrete optimal distinguishers efficiently based on the first algorithm's output.

We realize the automatic search of word-based division property with SMT method. We first show how to model division property propagations of basic operations by logical formulas. Moreover, by exclusion method, we construct formulas to depict the possible propagations calculated by Substitution rule. By setting initial division property and stopping rule rationally, the problem of searching division property can be transformed into an SMT problem, and we can efficiently search integral distinguishers with some openly available solvers.

As a result, we improve the previous integral distinguishers for SHACAL-2, LEA, CLEFIA, Rijndael-192, and Rijndael-256 according to the number of rounds. Moreover, the integral attacks for CLEFIA are improved by one round with the newly obtained distinguishers.

Discussion on the superiority to MILP method. We think it is hard to give a comprehensive comparison between MILP and SAT, and try to reflect the efficiency of SAT for ARX ciphers by recording the time spent on the search for the same distinguisher with a fixed initial division property under the same computation resource. The experimental results show that SAT model performs better than MILP model. As an illustration, for the optimal distinguisher of SHACAL-2, CryptoMiniSat returns the result after about 24 seconds, while MILP optimizer (Gurobi 7.0.2) takes about 44000 seconds, which is almost 1650 times as long as the SAT solver. Thus, it seems that SAT model is more suitable to search division properties for ARX ciphers.

Discussion on the optimality and completeness of the search. We confirm that the integral distinguishers are optimal under the search strategies defined in this paper. However, we cannot guarantee the completeness. If a more dedicated model for the modular addition is proposed, better integral distinguishers for ARX ciphers may be detected, which will be a future work.

Acknowledgements. The authors would like to thank the anonymous reviewers of Asiacrypt 2017 for their helpful comments. This work was supported by the 973 Program (No. 2013CB834205), NSFC Projects (No. 61572293), Science and Technology on Communication Security Laboratory of China (No. 9140c110207150c11050), as well as Chinese Major Program of National Cryptography Development Foundation (No. MMJJ20170102).

References

1. P. S. Barreto and V. Rijmen. The Whirlpool hashing function. In *First open NESSIE Workshop, Leuven, Belgium*, volume 13, page 14, 2000.
2. C. W. Barrett, R. Sebastiani, S. A. Seshia, and C. Tinelli. Satisfiability modulo theories. *Handbook of satisfiability*, 185:825–885, 2009.
3. R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers. The SIMON and SPECK lightweight block ciphers. In *Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, June 7-11, 2015*, pages 175:1–175:6, 2015.
4. E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. In *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, pages 2–21, 1990.
5. A. Biryukov, D. Khovratovich, and L. Perrin. Multiset-algebraic cryptanalysis of reduced Kuznyechik, Khazad, and secret SPNs. *IACR Trans. Symmetric Cryptol.*, 2016(2):226–247, 2016.
6. S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.
7. N. Courtois and G. V. Bard. Algebraic cryptanalysis of the data encryption standard. In *Cryptography and Coding, 11th IMA International Conference, Cirencester, UK, December 18-20, 2007, Proceedings*, pages 152–169, 2007.

8. J. Daemen and V. Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.
9. D. Dinu, L. Perrin, A. Udovenko, V. Velichkov, J. Großschädl, and A. Biryukov. Design strategies for ARX with provable bounds: Sparx and LAX. In *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, pages 484–513, 2016.
10. K. Fu, M. Wang, Y. Guo, S. Sun, and L. Hu. MILP-based automatic search algorithms for differential and linear trails for SPECK. In *Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers*, pages 268–288, 2016.
11. S. Galice and M. Minier. Improving integral attacks against Rijndael-256 up to 9 rounds. In *Progress in Cryptology - AFRICACRYPT 2008, First International Conference on Cryptology in Africa, Casablanca, Morocco, June 11-14, 2008. Proceedings*, pages 1–15, 2008.
12. D. Gerault, M. Minier, and C. Solnon. Constraint programming models for chosen key differential cryptanalysis. In *Principles and Practice of Constraint Programming - 22nd International Conference, CP 2016, Toulouse, France, September 5-9, 2016, Proceedings*, pages 584–601, 2016.
13. H. Handschuh and D. Naccache. SHACAL: a family of block ciphers. *Submission to the NESSIE project*, 2002.
14. D. Hong, J. Lee, D. Kim, D. Kwon, K. H. Ryu, and D. Lee. LEA: A 128-bit block cipher for fast encryption on common processors. In *Information Security Applications - 14th International Workshop, WISA 2013, Jeju Island, Korea, August 19-21, 2013, Revised Selected Papers*, pages 3–27, 2013.
15. D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B. Koo, C. Lee, D. Chang, J. Lee, K. Jeong, H. Kim, J. Kim, and S. Chee. HIGHT: A new block cipher suitable for low-resource device. In *Cryptographic Hardware and Embedded Systems - CHES 2006, 8th International Workshop, Yokohama, Japan, October 10-13, 2006, Proceedings*, pages 46–59, 2006.
16. A. A. Kamal and A. M. Youssef. Applications of SAT solvers to AES key recovery from decayed key schedule images. In *Fourth International Conference on Emerging Security Information Systems and Technologies, SECURWARE 2010, Venice, Italy, July 18-25, 2010*, pages 216–220, 2010.
17. L. R. Knudsen and D. Wagner. Integral cryptanalysis. In *Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4-6, 2002, Revised Papers*, pages 112–127, 2002.
18. S. Kölbl, G. Leander, and T. Tiessen. Observations on the SIMON block cipher family. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, pages 161–185, 2015.
19. Y. Li, W. Wu, and L. Zhang. Improved integral attacks on reduced-round CLEFIA block cipher. In *Information Security Applications - 12th International Workshop, WISA 2011, Jeju Island, Korea, August 22-24, 2011. Revised Selected Papers*, pages 28–39, 2011.
20. M. Matsui. Linear cryptanalysis method for DES cipher. In *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, pages 386–397, 1993.
21. M. Matsui. New block encryption algorithm MISTY. In *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings*, pages 54–68, 1997.

22. F. Mendel, C. Rechberger, M. Schläffer, and S. S. Thomsen. The rebound attack: Cryptanalysis of reduced Whirlpool and Grøstl. In *Fast Software Encryption, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22-25, 2009, Revised Selected Papers*, pages 260–276, 2009.
23. M. Minier, R. C. Phan, and B. Pousse. Distinguishers for ciphers and known key attack against Rijndael with large blocks. In *Progress in Cryptology - AFRICACRYPT 2009, Second International Conference on Cryptology in Africa, Gammarth, Tunisia, June 21-25, 2009. Proceedings*, pages 60–76, 2009.
24. N. Mouha and B. Preneel. Towards finding optimal differential characteristics for ARX: Application to Salsa20. Technical report, Cryptology ePrint Archive, Report 2013/328, 2013.
25. N. Mouha, Q. Wang, D. Gu, and B. Preneel. Differential and linear cryptanalysis using Mixed-Integer Linear Programming. In *Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011. Revised Selected Papers*, pages 57–76, 2011.
26. R. M. Needham and D. J. Wheeler. TEA extensions. *Report, Cambridge University, Cambridge, UK (October 1997)*, 1997.
27. PUB. FIPS 180-2: Secure hash standard (SHS). *US Department of Commerce, National Institute of Standards and Technology (NIST)*, 2012.
28. V. Rijmen and J. Daemen. Advanced Encryption Standard. *Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology*, pages 19–22, 2001.
29. N. Shibayama and T. Kaneko. A new higher order differential of CLEFIA. *IEICE Transactions*, 97-A(1):118–126, 2014.
30. Y. Shin, J. Kim, G. Kim, S. Hong, and S. Lee. Differential-linear type attacks on reduced rounds of SHACAL-2. In *Information Security and Privacy: 9th Australasian Conference, ACISP 2004, Sydney, Australia, July 13-15, 2004. Proceedings*, pages 110–122, 2004.
31. T. Shirai, K. Shibutani, T. Akishita, S. Moriai, and T. Iwata. The 128-bit blockcipher CLEFIA (extended abstract). In *Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg, March 26-28, 2007, Revised Selected Papers*, pages 181–195, 2007.
32. L. Song, Z. Huang, and Q. Yang. Automatic differential analysis of ARX block ciphers with application to SPECK and LEA. In *Information Security and Privacy - 21st Australasian Conference, ACISP 2016, Melbourne, VIC, Australia, July 4-6, 2016, Proceedings, Part II*, pages 379–394, 2016.
33. K. Specification. Specification of the 3GPP confidentiality and integrity algorithms. *Version*, 1:8–17.
34. N. Sugio, Y. Igarashi, and T. Kaneko. Integral characteristics of MISTY2 derived by division property. In *2016 International Symposium on Information Theory and Its Applications, ISITA 2016, Monterey, CA, USA, October 30 - November 2, 2016*, pages 151–155, 2016.
35. N. Sugio, Y. Igarashi, T. Kaneko, and K. Higuchi. New integral characteristics of KASUMI derived by division property. In *Information Security Applications - 17th International Workshop, WISA 2016, Jeju Island, Korea, August 25-27, 2016, Revised Selected Papers*, pages 267–279, 2016.
36. L. Sun, W. Wang, R. Liu, and M. Wang. MILP-aided bit-based division property for ARX-based block cipher. *IACR Cryptology ePrint Archive*, 2016:1101, 2016.
37. L. Sun, W. Wang, and M. Wang. MILP-aided bit-based division property for primitives with non-bit-permutation linear layers. *IACR Cryptology ePrint Archive*, 2016:811, 2016.

38. S. Sun, D. Gerault, P. Lafourcade, Q. Yang, Y. Todo, K. Qiao, and L. Hu. Analysis of AES, SKINNY, and others with constraint programming. *IACR Trans. Symmetric Cryptol.*, 2017(1):281–306, 2017.
39. S. Sun, L. Hu, P. Wang, K. Qiao, X. Ma, and L. Song. Automatic security evaluation and (related-key) differential characteristic search: Application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, pages 158–178, 2014.
40. Y. Todo. Integral cryptanalysis on full MISTY1. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, pages 413–432, 2015.
41. Y. Todo. Structural evaluation by generalized integral property. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, pages 287–314, 2015.
42. Y. Todo and M. Morii. Bit-based division property and application to SIMON family. In *Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers*, pages 357–377, 2016.
43. L. Wen and M. Wang. Integral zero-correlation distinguisher for ARX block cipher, with application to SHACAL-2. In *Information Security and Privacy - 19th Australasian Conference, ACISP 2014, Wollongong, NSW, Australia, July 7-9, 2014. Proceedings*, pages 454–461, 2014.
44. D. J. Wheeler and R. M. Needham. TEA, a tiny encryption algorithm. In *Fast Software Encryption: Second International Workshop. Leuven, Belgium, 14-16 December 1994, Proceedings*, pages 363–366, 1994.
45. 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 *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, pages 648–678, 2016.
46. H. Zhang and W. Wu. Structural evaluation for generalized feistel structures and applications to LBlock and TWINE. In *Progress in Cryptology - INDOCRYPT 2015 - 16th International Conference on Cryptology in India, Bangalore, India, December 6-9, 2015, Proceedings*, pages 218–237, 2015.
47. W. Zhang and V. Rijmen. Division cryptanalysis of block ciphers with a binary diffusion layer. *IACR Cryptology ePrint Archive*, 2017:188, 2017.
48. Y. Zheng, T. Matsumoto, and H. Imai. On the construction of block ciphers provably secure and not relying on any unproved hypotheses. In *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, pages 461–480, 1989.

A Integral Distinguishers of Whirlpool and Rijndael

A.1 Integral Distinguishers of Whirlpool

Note the intermediate state of the internal block cipher of Whirlpool can be represented by an 8×8 matrix of bytes, and the indexes of the involved bytes are illustrated in Fig. 3.

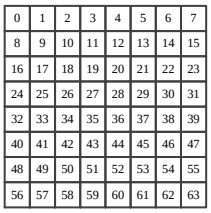

Fig. 3: Indexes for Whirlpool.

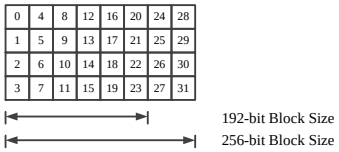

Fig. 4: Indexes for Rijndael-192 and Rijndael-256.

The integral distinguishers obtained in the paper are illustrated as follows.

$$\text{Active Bytes: } \{0\} \xrightarrow{\text{3 Rounds}} \text{Zero-sum Bytes: } \{0-63\},$$

$$\text{Active Bytes: } \{0, 22, 29, 36, 43, 50, 57\} \xrightarrow{\text{4 Rounds}} \text{Zero-sum Bytes: } \{0-63\},$$

$$\text{Active Bytes: } \{0-5, 8-12, 15-19, 22-26, 29-33, 36-40, 43-47, 50-55, 57-62\} \xrightarrow{\text{5 Rounds}} \text{Zero-sum Bytes: } \{0-63\}.$$

A.2 Integral Distinguishers of Rijndael

For Rijndael family of block ciphers, the internal state can be treated as a $4 \times N_b$ matrix of bytes, where N_b is the number of 32-bit words in the block. The indexes for the matrix is shown in Fig. 4.

The integral distinguishers for Rijndael-192 mentioned in the paper are listed as follows:

$$\text{Active Bytes: } \{0\} \xrightarrow{\text{3 Rounds}} \text{Zero-sum Bytes: } \{0-23\},$$

$$\text{Active Bytes: } \{0, 5, 10\} \xrightarrow{\text{4 Rounds}} \text{Zero-sum Bytes: } \{16-23\},$$

$$\text{Active Bytes: } \{0, 4, 5, 9, 10, 14, 15, 19\} \xrightarrow{\text{5 Rounds}} \text{Zero-sum Bytes: } \{12-19\},$$

$$\text{Active Bytes: } \{0-7, 9-12, 14-17, 19-22\} \xrightarrow{\text{6 Rounds}} \text{Zero-sum Bytes: } \{0-7\}.$$

And the distinguishers we found for Rijndael-256 are presented below:

$$\text{Active Bytes: } \{0\} \xrightarrow{\text{3 Rounds}} \text{Zero-sum Bytes: } \{0-31\},$$

$$\text{Active Bytes: } \{0, 5\} \xrightarrow{\text{4 Rounds}} \text{Zero-sum Bytes: } \{8-11, 20-31\},$$

$$\text{Active Bytes: } \{0, 5, 14, 19\} \xrightarrow{\text{5 Rounds}} \text{Zero-sum Bytes: } \{8-11, 24-27\},$$

$$\text{Active Bytes: } \{0, 3-5, 8, 9, 12-14, 16-19, 21-23, 26, 27, 30, 31\} \xrightarrow{\text{6 Rounds}} \text{Zero-sum Bytes: } \{8-11, 24-27\}.$$