MILP-Based Automatic Search Algorithms for Differential and Linear Trails for Speck

Kai Fu[1], Meiqin Wang*[1,2], Yinghua Guo[1], Siwei Sun[3,4], Lei Hu[3,4]

[1] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan 250100, China
[2] State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China
[3] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
[4] Data Assurance and Communication Security Research Center, Chinese Academy of Sciences, Beijing 100093, China

Abstract. In recent years, Mixed Integer Linear Programming (MILP) has been successfully applied in searching for differential characteristics and linear approximations in block ciphers and has produced the significant results for some ciphers such as SIMON (a family of lightweight and hardware-optimized block ciphers designed by NSA) etc. However, in the literature, the MILP-based automatic search algorithm for differential characteristics and linear approximations is still infeasible for block ciphers such as ARX constructions. In this paper, we propose an MILP-based method for automatic search for differential characteristics and linear approximations in ARX ciphers. By researching the properties of differential characteristic and linear approximation of modular addition in ARX ciphers, we present a method to describe the differential characteristic and linear approximation with linear inequalities under the assumptions of independent inputs to the modular addition and independent rounds. We use this representation as an input to the publicly available MILP optimizer Gurobi to search for differential characteristics and linear approximations for ARX ciphers. As an illustration, we apply our method to Speck, a family of lightweight and software-optimized block ciphers designed by NSA, which results in the improved differential characteristics and linear approximations compared with the existing ones. Moreover, we provide the improved differential attacks on Speck48, Speck64, Speck96 and Speck128, which are the best attacks on them in terms of the number of rounds.

Keywords: Automatic Search, Differential Characteristic, Linear Approximation, ARX, Speck

* Corresponding author.

1 Introduction

Differential attacks [3] and linear attacks [15] are the most fundamental cryptanalytic methods. They have been used in the cryptanalysis of numerous symmetric ciphers. Since the first and most important thing for the two methods is to identify differential characteristics and linear approximations, the automatic search algorithms for differential characteristics and linear approximations have been a focus of cryptographer's concern. At EUROCRYPT'94, Matsui [16] presented the branch-and-bound search algorithm and found the differential characteristics and linear approximations for DES block cipher. The branch-and-bound search algorithm is one of the most powerful and classic search tools and is still widely used now. Another research line for the application of automatic search algorithm is to provide the provable security against differential cryptanalysis and linear cryptanalysis, which is usually achieved by automatic searching for the minimal number of active S-boxes.

Mixed-Integer Linear Programming (MILP) has been explicitly applied in constructing automatic search algorithm in differential and linear cryptanalysis. The problem of MILP is a class of optimization problems derived from Linear Programming which aims to optimize an objective function under certain constrains. Mouha *et al.* [18] and Wu *et al.* [28] translated the problem of counting the minimal number of differentially active S-boxes to an MILP problem which can be solved automatically with open source or commercially available MILP solvers. Their method has been applied in searching for the differential and linear characteristics with specific patterns [14,29] and counting the minimal number of active S-boxes of bit-oriented block ciphers by introducing bit-level representations [21, 27].

Recently, the MILP-based method has been developed to be a general method to automatically search for the real differential characteristics. Sun *et al.* [22] constructed the MILP-based model to search for (related-key) differential characteristics by generating linear inequalities from the differential distribution table of S-box, where only partial linear inequalities are used in MILP model to make it solvable in practical time. Their search algorithm, however, is heuristic, since the identified differential characteristics may not be consistent. By computing a small number of inequalities which can exactly describe the differential distribution table of an S-box with the greedy algorithm, Sun *et al.* [24] transformed the heuristic searching method to the exact and practical searching method. Moreover, they constructed the MILP-based model for automatically searching for linear approximations and extended these models to search for differential and linear hull. Sun *et al.*'s method [22,24] is applicable to block ciphers involving bitwise XOR, S-box operation and the linear layer with bit permutation[5]. Although the general linear layer can be transformed into bit XOR operations, it makes the MILP problem much more difficult to be

[5] Although SIMON has no S-box, the And and XOR operations for SIMON could be regarded as one S-box. So they also applied the method to SIMON.

solved in practical time since more XOR operations result in more variables and constraints.

Due to the excellent performance of ARX-based ciphers in software, many symmetric-key ciphers are designed based on ARX operations[6]. It is worth noting that the cryptanalytic techniques for ARX ciphers are very different from those for ciphers with S-boxes such as AES and DES. In particular, the search algorithms for differential characteristics and linear approximations for ARX cipher utilize the different principle compared with those for ciphers with S-boxes. In [10, 12, 17], the methods of automatic search for differential characteristics in ARX designs are provided, but the methods are only compatible with ARX-based Hash functions where the key is known and can be freely chosen. By using the partial differential distribution table and Matsui's branch-and-bound algorithm, Biryukov and Velichkov [4] presented the first automatic search algorithm for differential characteristics in ARX block ciphers, such as (X)TEA and Speck. In a very recent paper [5] appearing in this volume of FSE'16, Biryukov *et al.* proposed the first adaptation of Matsui's algorithm for finding the best differential and linear trails in ARX ciphers.

Although MILP-based search algorithm has got extremely remarkable application for some block ciphers, the current method cannot be applied to ARX block ciphers. A straightforward method to apply MILP model for ARX constructions is to regard the modular addition in $\mathbb{F}_2^n$ as a $2n \times n$ S-box and compute a small number of linear inequalities to exactly represent the differential or linear pattern of the modular addition. However, in this way the number of linear inequalities is too large to be solved in practical time for real ARX ciphers where n is typically at least 16. This motivates us to study MILP-based search method for ARX block ciphers.

1.1 Our Contributions.

In this paper, we revisit the differential property and linear property for modular addition and provide a new framework of constructing the MILP model. Concretely, we transform the differential property of modular addition shown in [13] into linear inequalities to describe all possible differential patterns and the corresponding differential probabilities. Moreover, we use linear inequalities to capture all possible linear patterns and the corresponding correlations based on the automaton algorithm for correlation of modular addition in [19, 25]. The number of the resulting linear inequalities is significantly less than that of linear inequalities produced by regarding modular addition as one S-box. With the linear inequalities, we can construct the MILP model to automatically search for differential characteristics and linear approximations using the the commercial optimizer Gurobi, where the object function is the probability of differential characteristic or the correlation of linear approximation.

During constructing MILP models, we assume that the two inputs to modular addition and the consecutive rounds are independent. However, as demonstrated

[6] ARX operation: modular addition, bit rotation and XOR.

in [26], for some ARX constructions, the inputs to modular addition and the consecutive rounds are not independent, which will result that the practical probability (resp. correlation) of our identified differential (resp. linear) tails for some fixed key may vary significantly from that derived from our model. This deviation will have effect on the success rate of the attacks from practitioner's perspective.

As an illustration, we apply our method to the block cipher Speck, which is a family of lightweight block ciphers publicly released by the National Security Agency (NSA) and has been optimized for performance in software implementations [2]. A variety of block sizes and key sizes for different implementations are provided for it. Since its publication, Speck has received much attention and many cryptanalytic results have been given. Abed et al. presented differential and rectangle attacks for almost all variants of Speck [1]. At FSE'14 [6], Biryukov *et al.* searched for the differential characteristics, which cover 9, 11 and 14 rounds for Speck32, Speck48 and Speck64, respectively, and are better than the differential characteristics in [1]. In [9], Dinur proposed the sub-cipher attack and improved the key recovery attacks on all variants of Speck using the differential characteristics in [6]. In [5], Biryukov *et al.* presented the probabilities of the best differential trails for up to 10, 9, 8, 7, and 6 rounds of Speck32, Speck48, Speck64, Speck96 and Speck128 respectively and evaluate the security bounds of Speck against single-trail differential cryptanalysis under the Markov assumption. As regards to linear cryptanalysis, Yao *et al.* identified 9, 9, 12, 6 and 6 rounds linear approximations for Speck32, Speck48, Speck64, Speck96 and Speck128, respectively [30], and gave the key recovery attacks.

We use our models to search for the differential and linear trails for Speck. In order for the MILP tool to run in reasonable time for larger block sizes ($>$ 48 bits), we split the block cipher into two or three parts – upper (middle) and lower. We then search for trails independently in each part, by ensuring that the output difference (mask) for one part is the same as the input difference (mask) for its following part. For Speck48, Speck64, Speck96 and Speck128, we find better differential characteristics and linear approximations than those of previous works under the assumptions of independent inputs to the modular addition and independent rounds. With the new differential characteristics, we improve the differential attacks on the four variants of Speck. Comparing with the previous best attacks for them [9], we can attack one, one, three and five more rounds for Speck48, Speck64, Speck96 and Speck128 with any key size, respectively. We summarize known attacks on Speck in Table 1. We compare our identified differential characteristics and linear approximations with those of previous works in Table 2.

Outline. The remainder of this paper is organized as follows. Section 2 gives a brief description of the existing MILP-based search methods for block cipher. Section 3 and Section 4 introduce MILP-based algorithm for automatic searching for differential characteristics and linear approximations for ARX ciphers. We apply the new search tools in Speck and give the improved differential attacks

Table 1: Summary of Attacks on Speck

Variant $2n/mn$	Rounds Attacked/ Total Rounds	Time	Data	Memory	Method	Ref.
48/72	11/22	$2^{67.93}$	$2^{43.727}$	-	Linear	[30]
	12/22	$2^{58.8}$	$2^{43.2}$	$2^{45.8}$	Rectangle	[1]
	12/22	$2^{45.3}$	2^{45}	2^{24}	Differential	[1]
	14/22	2^{65}	2^{41}	2^{22}	Differential	[9]
	15/22	$\mathbf{2^{70}}$	$\mathbf{2^{46}}$	$\mathbf{2^{22}}$	**Differential**	**This Paper**
48/96	12/23	$2^{91.93}$	$2^{43.727}$	-	Linear	[30]
	12/23	$2^{58.8}$	$2^{43.2}$	$2^{45.8}$	Rectangle	[1]
	12/23	$2^{45.3}$	2^{45}	2^{24}	Differential	[1]
	15/23	2^{89}	2^{41}	2^{22}	Differential	[9]
	16/23	$\mathbf{2^{94}}$	$\mathbf{2^{46}}$	$\mathbf{2^{22}}$	**Differential**	**This Paper**
64/96	14/26	$2^{94.9}$	$2^{62.7}$	-	Linear	[30]
	14/26	$2^{89.4}$	$2^{63.6}$	$2^{65.6}$	Rectangle	[1]
	15/26	$2^{61.1}$	2^{61}	2^{32}	Differential	[1]
	18/26	2^{93}	2^{61}	2^{22}	Differential	[9]
	19/26	$\mathbf{2^{95}}$	$\mathbf{2^{63}}$	$\mathbf{2^{22}}$	**Differential**	**This Paper**
64/128	15/27	$2^{126.9}$	$2^{62.7}$	-	Linear	[30]
	14/27	$2^{89.4}$	$2^{63.6}$	$2^{65.6}$	Rectangle	[1]
	15/27	$2^{61.1}$	2^{61}	2^{32}	Differential	[1]
	19/27	2^{125}	2^{61}	2^{22}	Differential	[9]
	20/27	$\mathbf{2^{127}}$	$\mathbf{2^{63}}$	$\mathbf{2^{22}}$	**Differential**	**This Paper**
96/96	8/28	$2^{74.7}$	$2^{27.6}$	-	Linear	[30]
	15/28	$2^{89.1}$	2^{89}	2^{48}	Differential	[1]
	16/28	2^{85}	2^{85}	2^{22}	Differential	[9]
	19/28	$\mathbf{2^{88}}$	$\mathbf{2^{88}}$	$\mathbf{2^{22}}$	**Differential**	**This Paper**
96/144	9/29	$2^{122.7}$	$2^{27.6}$	-	Linear	[30]
	16/29	$2^{135.9}$	$2^{90.9}$	$2^{94.5}$	Rectangle	[1]
	15/29	$2^{89.1}$	2^{89}	2^{48}	Differential	[1]
	17/29	2^{133}	2^{85}	2^{22}	Differential	[9]
	20/29	$\mathbf{2^{136}}$	$\mathbf{2^{88}}$	$\mathbf{2^{22}}$	**Differential**	**This Paper**
128/128	8/32	$2^{92.7}$	$2^{28.3}$	-	Linear	[30]
	16/32	$2^{111.1}$	2^{116}	2^{64}	Differential	[1]
	17/32	2^{113}	2^{113}	2^{22}	Differential	[9]
	22/32	$\mathbf{2^{120}}$	$\mathbf{2^{120}}$	$\mathbf{2^{22}}$	**Differential**	**This Paper**
128/192	9/33	$2^{156.7}$	$2^{28.3}$	-	Linear	[30]
	16/33	$2^{111.1}$	2^{116}	2^{64}	Differential	[1]
	18/33	$2^{182.7}$	$2^{125.9}$	$2^{121.9}$	Rectangle	[1]
	18/33	2^{177}	2^{113}	2^{22}	Differential	[9]
	23/33	$\mathbf{2^{184}}$	$\mathbf{2^{120}}$	$\mathbf{2^{22}}$	**Differential**	**This Paper**
128/256	7/34	$2^{220.7}$	$2^{28.3}$	-	Linear	[30]
	16/34	$2^{111.1}$	2^{116}	2^{64}	Differential	[1]
	18/34	$2^{182.7}$	$2^{125.9}$	$2^{121.9}$	Rectangle	[1]
	19/34	2^{241}	2^{113}	2^{22}	Differential	[9]
	24/34	$\mathbf{2^{248}}$	$\mathbf{2^{120}}$	$\mathbf{2^{22}}$	**Differential**	**This Paper**

Table 2: Summary of Differential Characteristics and Linear Approximations for Speck

Cipher	Differential Characteristic			Linear Approximation		
	# Rounds	$\log_2 p$	Ref.	# Rounds	$\log_2 c$	Ref.
Speck32	9	-31	[1]	9	-14	[30]
	9	-30	[6]	9	-14	This paper
	9	-30	This paper			
Speck48	10	-41	[1]	9	-20	[30]
	11	-47	[6]	10	-22	This paper
	11	-45	This paper			
Speck64	13	-59	[1]	11	-25	[30]
	13	-51	This paper	12	-31	[30]
	14	-60	[6]	13	-30	This paper
	14	-56	This paper			
	15	-62	This paper			
Speck96	13	-84	[1]	6	-11	[30]
	13	-67	This paper	15	-45	This paper
	16	-87	This paper			
Speck128	14	-112	[1]	6	-11	[30]
	14	-90	This paper	16	-58	This paper
	19	-119	This paper			

on all variants of Speck except Speck32 in Section 5. Finally, we conclude the paper in Section 6.

2 Sun *et al.'s* MILP-Based Automatic Search for (Related-Key) Differential and Linear Trails (Hull)

In this section, we briefly recall Sun *et al.*'s algorithm. For more details of their algorithm, we refer to [22, 24].

Objective Function of Differential Model. Let x_i denote the difference variable for the bit i. That is, $x_i = 0$ if there is no difference at bit i; Otherwise, $x_i = 1$. Another bit variable A_j is used to denote the activity of an S-box, *i.e.*, $A_j = 0$ if the S-box is non-active; Otherwise, $A_j = 1$. The objective function is to minimize the sum of all variables $\sum_j A_j$, which indicates the activities of the S-boxes appearing in the schematic description of the encryption and key schedule algorithm.

Constraints of Differential Model. For every XOR operation with bit-level input differences a, b and bit-level output difference c, the constraints include

$$\begin{cases} d_\oplus \geq a, d_\oplus \geq b, d_\oplus \geq c \\ a + b + c \geq 2d_\oplus \\ a + b + c \leq 2 \end{cases} \tag{1}$$

where $d_\oplus$ is a dummy bit variable.

Next, we describe the constraints of the differential properties of an S-box in a more accurate way. For an $\omega \times \nu$ S-box denoted by A_t, the input and output differences are $(x_0, \ldots, x_{\omega-1})$ and $(y_0, \ldots, y_{\nu-1})$, respectively. Then

$$\begin{cases} A_t - x_k \geq 0, k \in \{0, \ldots, \omega - 1\} \\ -A_t + \sum_{j=0}^{\omega-1} x_j \geq 0 \end{cases} \tag{2}$$

which ensures that nonzero input difference must activate the S-box.

Let $(x_0, \ldots, x_{\omega-1}, y_0, \ldots, y_{\nu-1}) \in \{0,1\}^{\omega+\nu} \subseteq \mathbb{R}^{\omega+\nu}$ denote an $(\omega + \nu)$-dimensional vector, where $\mathbb{R}$ is the real number field. By computing the H-Representation of the convex hull of all possible input-output differential patterns of an S-box, many linear inequalities which can be used to remove some impossible differential patterns of the S-box are obtained. The greedy algorithm in [24] is applied to select a subset of the H-Representation of the convex hull with less inequalities. As a result, they generate only a small number of linear inequalities, which can be used to exactly describe the differential pattern of S-box and construct the MILP problem. Using any MILP optimizer such as Gurobi [11], good differential characteristics can be found. If we set the value of the object function as N, finish the solving process and output the current solution till the value of object function is reduced to N. The corresponding solution is the identified differential characteristic with N active S-boxes.

Note that this exact searching method is also applicable to searching for the linear approximations.

Objective Function of Linear Model. Some notations for differential model are also used in linear model, e.g., A_j denotes the activity of an S-box and the objective function is to minimize $\sum_j A_j$.

Constraints of Linear Model. For every XOR operation with input masks a, b and output mask c, the constraints should be

$$a = b = c.$$

For every three-forked branch with input mask a and output masks b and c, the constraints should be

$$\begin{cases} d_\curlywedge \geq a, d_\curlywedge \geq b, d_\curlywedge \geq c \\ a + b + c \geq 2d_\curlywedge \\ a + b + c \leq 2 \end{cases} \tag{3}$$

where $d_\curlywedge$ is a dummy bit variable.

For an $\omega \times \nu$ S-box denoted by A_t, the input and output masks are $(x_0, \ldots, x_{\omega-1})$ and $(y_0, \ldots, y_{\nu-1})$, respectively. If the output mask is nonzero, $A_t = 1$; Otherwise, $A_t = 0$. Then, we have

$$\begin{cases} A_t - y_k \geq 0, k \in \{0, \ldots, \nu - 1\} \\ -A_t + \sum_{j=0}^{\nu-1} y_j \geq 0 \end{cases}$$

which ensures that nonzero output mask must activate the S-box.

For an $(\omega+\nu)$-dimensional vector $(x_0,\ldots,x_{\omega-1},y_0,\ldots,y_{\nu-1}) \in \{0,1\}^{\omega+\nu} \subseteq \mathbb{R}^{\omega+\nu}$, compute a small number of linear inequalities to exactly represent the linear pattern of S-box. The other processes are similar to those in the model of searching for differential characteristics.

In addition, the technique has been extended to find differential or linear hull [24]. By adding the constraints imposed by the given properties (such as fixed difference or linear mask), they updated the MILP model and obtained all trails which consist of the given differential or linear hull.

3 MILP-Based Algorithm for Automatic Search for Differential Characteristics in ARX Ciphers

In this section, we analyze the differential characteristics of modular addition and identify important properties, which are crucial to the construction of our MILP-based models for ARX ciphers. Using our method, we can give the linear inequalities which can exactly describe all differential patterns for modular addition.

3.1 XOR-Differential Characteristics of Modular Addition

Definition 1. *Let α,β and γ be fixed n-bit `XOR` differences. The `XOR`-differential probability (`DP`) of addition modulo 2^n (xdp^+) is the probability with which α and β propagate to γ through the `ADD` operation, computed over all pairs of n-bit inputs (x,y):*

$$xdp^+(\alpha,\beta\rightarrow\gamma) = 2^{-2n}\cdot\#\{(x,y):((x\oplus\alpha)+(y\oplus\beta))\oplus(x+y)=\gamma\}.$$

In [13], Lipmaa *et al.* showed Algorithm 2 to compute $xdp^+(\alpha,\beta\rightarrow\gamma)$ which consists of two steps: the first step is to verify if the differential characteristic is possible and the second step is to compute the differential probability xdp^+. More precisely, the above two steps are shown in Theorem 1 and Theorem 2, respectively.

Theorem 1 (see [13]). *The differential $(\alpha,\beta\rightarrow\gamma)$ is possible iff $(\alpha[0]\oplus\beta[0]\oplus\gamma[0]) = 0$ and $\alpha[i-1]=\beta[i-1]=\gamma[i-1]=\alpha[i]\oplus\beta[i]\oplus\gamma[i]$ for $\alpha[i-1]=\beta[i-1]=\gamma[i-1]$, $i\in[1,n-1]$.*

Theorem 2 (see [13]). *Assume that $(\alpha,\beta\rightarrow\gamma)$ is a possible differential characteristic, then the differential probability $xdp^+ = 2^{-\sum_{i=0}^{n-2}\neg eq(\alpha[i],\beta[i],\gamma[i])}$, where*

$$eq(\alpha[i],\beta[i],\gamma[i]) = \begin{cases} 1 & \alpha[i]=\beta[i]=\gamma[i] \\ 0 & others \end{cases}.$$

Theorem 1 can be used to decide if the differential characteristic $(\alpha, \beta \rightarrow \gamma)$ for modular addition is possible. For instance, the differential $(\alpha, \beta \rightarrow \gamma) = (11100, 11100 \rightarrow 11110)$ is impossible as $\alpha[0] = \beta[0] = \gamma[0] \neq \alpha[1] \oplus \beta[1] \oplus \gamma[1]$. Using Theorem 2, the probability of the differential characteristic can be computed efficiently. For example, for the differential $(\alpha, \beta \rightarrow \gamma) = (11100,\ 00110 \rightarrow 10110)$, the probability $xdp^+(\alpha, \beta \rightarrow \gamma) = 2^{-(\neg eq(0,0,0)+\neg eq(0,1,1)+\neg eq(1,1,1)+\neg eq(1,0,0))} = 2^{-2}$.

From Theorem 2, if the n-bit differential characteristic is possible, the probability is only related with $(\alpha[i], \beta[i], \gamma[i])$ for $i \in [0, n-2]$. Taking advantage of this property, we can construct the MILP model to compute the differential probability xdp^+. More details are shown in the following.

3.2 MILP Model for Differential Characteristics of Modular Addition

In order to append the first condition $\alpha[0] \oplus \beta[0] \oplus \gamma[0] = 0$ in Theorem 1 to the set of the linear inequalities, we derive five linear inequalities satisfying the first condition. The five linear inequalities are listed as follows,

$$\begin{cases} d_\oplus \geq \alpha[0], d_\oplus \geq \beta[0], d_\oplus \geq \gamma[0] \\ \alpha[0] + \beta[0] + \gamma[0] - 2d_\oplus \geq 0 \\ \alpha[0] + \beta[0] + \gamma[0] \leq 2 \end{cases} \tag{4}$$

where $d_\oplus$ is a dummy bit variable.

Let the vector $(\alpha[i], \beta[i], \gamma[i],\ \alpha[i+1], \beta[i+1], \gamma[i+1])$ denote the relation of the differential values for the i-th and the $(i+1)$-th bits. We have that there are totally 56 possible patterns for the vector in accordance with Theorem 1. For example, the differential pattern $(0, 0, 0, 1, 1, 1)$ is impossible as $\alpha_i = \beta_i = \gamma_i \neq \alpha_{i+1} \oplus \beta_{i+1} \oplus \gamma_{i+1}$. Moreover, in order to compute the differential probability efficiently, we append $\neg eq(\alpha[i], \beta[i], \gamma[i])$ to the vector. As described in [23], using the **inequality_generator**() function in the **sage**. **geometry**. **polyhedron** class of the SAGE Computer Algebra System [20], we get 65 linear inequalities satisfying all 56 possible patterns. Based on the greedy algorithm in [24], the number of linear inequalities can be reduced from 65 to 13. Furthermore, the 13 linear inequalities can be used to compute the probability of $(\alpha[i] \| \beta[i] \| \gamma[i] \rightarrow \alpha[i+1] \| \beta[i+1] \| \gamma[i+1])$ as the variable $\neg eq(\alpha[i], \beta[i], \gamma[i])$ is involved.

Actually, the 13 linear inequalities only represent the second condition $\alpha[i] = \beta[i] = \gamma[i] = \alpha[i+1] \oplus \beta[i+1] \oplus \gamma[i+1],\ i \in [0, n-2]$ in Theorem 1. In total, there are $(13 \times (n-1) + 5)$ linear inequalities to represent the differential property of addition modulo 2^n with two inputs of n-bit length. As described in Theorem 2, the differential probability $xdp^+ = 2^{-\sum_{i=0}^{n-2} \neg eq(\alpha[i], \beta[i], \gamma[i])}$.

3.3 MILP Model for Differential Characteristics of ARX Ciphers

Besides modular addition, the XOR operations, three-forked branch and the circular shift operations are also involved in ARX ciphers. For each XOR

operation, we can also use Inequalities (4). For each three-forked branch operation with input differences a, b and output difference c, the constraints should be

$$a = b = c.$$

For the case of circular shift, we can also list some equations for the related bits. So far, we have finished the construction of all linear inequalities or equations for each operations in ARX ciphers which compose the **constraints** of MILP model for differential characteristics of ARX ciphers.

As the differential probability $xdp^+ = 2^{-\sum_{i=0}^{n-2} \neg eq(\alpha[i],\beta[i],\gamma[i])}$ can be computed using the method described in Section 3.2, we set the **objective function** for the r-round differential characteristic as the $\sum_{j=1}^{r}\sum_{i=0}^{n-2} \neg eq(\alpha_j[i], \beta_j[i], \gamma_j[i])$ where α_j, β_j and γ_j are the input differences and output difference of modular addition for the j-th round. We aim to find the minimal value of $\sum_{j=1}^{r}\sum_{i=0}^{n-2} \neg eq(\alpha_j[i], \beta_j[i], \gamma_j[i])$ which represents the differential probability of the best identified differential characteristic. We can use the Gurobi optimizer to solve the system of inequalities to search for differential characteristics for ARX ciphers. However, just being able to obtain one differential characteristic may be not enough. We can apply Sun's method [24] to our new MILP model and find the differential of ARX ciphers.

Note that in the above model, we assume that the two inputs to modular addition and the consecutive rounds are independent. However, for some ARX constructions, they are not independent, which will result that the practical probability of our identified differential characteristics for some fixed key may vary significantly from that derived from our model.

4 MILP Models for Automatic Search for Linear Approximations in ARX Ciphers

In this section, we revisit the property of linear approximations for modular addition operation and develop a new MILP-based tool to search for the linear approximations for ARX ciphers.

4.1 Linear Approximations for Modular Addition

Let n be a non-negative integer. Given two vectors $x = (a_{n-1}, \ldots, a_0)$ and $y = (b_{n-1}, \ldots, b_0) \in \mathbb{F}_2^n$, let $x \cdot y$ denote the standard inner product $x \cdot y = a_{n-1}b_{n-1} \oplus \cdots \oplus a_0 b_0$.

Definition 2. *Let Λ_α, Λ_β and Γ be fixed n-bit linear masks. The correlation of addition modulo 2^n ($cor_\boxplus$) with input masks $\Lambda_\alpha, \Lambda_\beta$ and output mask Γ can be computed over all pairs of n-bit inputs (x, y):*

$$\begin{aligned} &cor_\boxplus(\Gamma, \Lambda_\alpha, \Lambda_\beta) = cor(\Gamma \cdot (x+y) \oplus \Lambda_\alpha \cdot x \oplus \Lambda_\beta \cdot y) \\ &= 2^{-2n}(\#\{x, y \in \mathbb{F}_2^n : \Gamma \cdot (x+y) \oplus \Lambda_\alpha \cdot x \oplus \Lambda_\beta \cdot y = 0\} \\ &- \#\{x, y \in \mathbb{F}_2^n : \Gamma \cdot (x+y) \oplus \Lambda_\alpha \cdot x \oplus \Lambda_\beta \cdot y = 1\}). \end{aligned}$$

Based on a fairly simple classification of the linear approximations of the carry function, Nyberg and Wallén derive an efficient algorithm for computing the correlation of linear approximation of addition modulo 2^n with k inputs in [19,25]. Since we only consider modular addition with two inputs, we describe this method only for two inputs in [19,25] as follows.

Theorem 3 (see [19,25]). *For the linear approximation of addition modulo 2^n of two inputs with input masks $\Lambda_\alpha, \Lambda_\beta$ and output mask Γ, $\Lambda_\alpha, \Lambda_\beta, \Gamma \in \mathbb{F}_2^n$ and $\Lambda_\alpha = (\Lambda_\alpha[n-1], \ldots, \Lambda_\alpha[0])$, $\Lambda_\beta = (\Lambda_\beta[n-1], \ldots, \Lambda_\beta[0])$, $\Gamma = (\Gamma[n-1], \ldots, \Gamma[0])$, we define the vector $u = (u[n-1], \ldots, u[0])$ where $u[i] = 4\Gamma[i]+2\Lambda_\alpha[i]+\Lambda_\beta[i], 0 \leq u[i] < 8, 0 \leq i < n$. The correlation can be computed with the following linear representation,*

$$cor_{\boxplus}(\Gamma, \Lambda_\alpha, \Lambda_\beta) = LA_{u[n-1]}A_{u[n-2]} \cdots A_{u[1]}A_{u[0]}C, \tag{5}$$

where $A_r, r = 0, \ldots, 7$, is 2×2 matrice,

$$A_0 = \frac{1}{2}\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, A_1 = A_2 = -A_4 = \frac{1}{2}\begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix},$$

$$A_7 = \frac{1}{2}\begin{bmatrix} 0 & 2 \\ 1 & 0 \end{bmatrix}, -A_3 = A_5 = A_6 = \frac{1}{2}\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix},$$

L is a row vector $L = (1\ 0)$,and C is a column vector $C = (1\ 1)^T$.

For example, for the linear approximation with binary vector masks ($\Gamma = 10100, \Lambda_\alpha = 11110, \Lambda_\beta = 11000$), $u = 73620_8$ and $cor_{\boxplus}$=$LA_7A_3A_6A_2A_0C = -2^{-3}$.

In order to provide a fast implementation for Theorem 3, Nyberg and Wallén utilized the automaton to calculate $LA_{u[n-1]}A_{u[n-2]} \cdots A_{u[1]}A_{u[0]}C$ by multiplying from left to right. Let $e_0 = L = (1\ 0)$ and $e_1 = (0\ 1)$, then we can show the state transitions in Fig. 1. When reading u from left to right, if the automaton ends up in state 0, then $LA_{u[n-1]}A_{u[n-2]} \cdots A_{u[1]}A_{u[0]}C = 0$. If the automaton ends up in state e_0 or e_1, then $LA_{u[n-1]}A_{u[n-2]} \cdots A_{u[1]}A_{u[0]}C = \pm 2^{-t}$, where t is the number of transitions marked by a solid arrow, and the sign is determined by the number of occurrences of $\{3, 4\}$:

$$LA_{u[n-1]}A_{u[n-2]} \cdots A_{u[1]}A_{u[0]}C > 0$$

if and only if the number of occurrences is even. For example, as $u = 73620_8$, $LA_7A_3A_6A_2A_0C = -2^{-3}$.

Based on the Picture 1, we will give Proposition 1 to calculate the absolute value of the correlation $cor_{\boxplus}(\Gamma, \Lambda_\alpha, \Lambda_\beta)$.

Proposition 1. *For the linear approximation of addition modulo 2^n of two inputs with input masks $\Lambda_\alpha, \Lambda_\beta$ and output mask Γ, the state transitions of the automaton are shown in Fig. 2, where $u[i] = 4\Gamma[i] + 2\Lambda_\alpha[i] + \Lambda_\beta[i], 0 \leq u[i] < 8, 0 \leq i < n$ and $\epsilon_j \in \{e_0, e_1\}, 0 \leq j < n$. If the correlation for the linear approximation is non-zero, the absolute value of the correlation can be computed as follows,*

$$|cor_{\boxplus}(\Gamma, \Lambda_\alpha, \Lambda_\beta)x| = 2^{-\#\{0<i<n|\epsilon_i=e_1\}}. \tag{6}$$

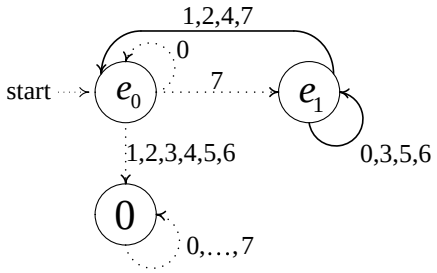

Fig. 1: State Transitions for $u = 73620_8$

Fig. 2: State Transitions for Addition Modulo 2^n

Based on Proposition 1, we construct the MILP model to compute the absolute value of correlation of modular addition with two inputs in the following.

4.2 MILP Model for Linear Approximations of Modular Addition

In this part, we will introduce a method to describe linear property of modular addition in Theorem 3 and Proposition 1 as linear inequalities.

For the state transition from ϵ_{i+1} to ϵ_i under $u[i]$, $0 \leq i < n$, the bit variable s_i is defined as follows, $s_i = 0$ if $\epsilon_i = e_0$, and $s_i = 1$ if $\epsilon_i = e_1$. We utilize the vector $(s_{i+1}, \Gamma[i], \Lambda_\alpha[i], \Lambda_\beta[i], s_i)$ to denote the state transition, so $e_{s_{i+1}} A_{u[i]} = e_{s_i}$. For the vector $(s_{i+1}, \Gamma[i], \Lambda_\alpha[i], \Lambda_\beta[i], s_i)$, there are only 10 possible transitions for the vector. As described in Section 3.2, we also use the **inequality_generator**() function in the **sage**. **geometry**. **polyhedron** class of SAGE and the greedy algorithm in [24] to get eight linear inequalities satisfying all 10 possible transitions. Note that there is an additional constraint $\epsilon_n = e_0$ according to Fig. 2. Hence, for linear approximation of addition modulo 2^n with two inputs, the constraints contain $8 \times n + 1$ linear inequalities and the absolute value of correlation is $|cor_{\boxplus}| = 2^{-\sum_{i=1}^{n-1} s_i}$.

4.3 MILP Model for Linear Approximations for ARX Ciphers

For each XOR, three-forked branch and circular shift operations, we can also use the method in Section 2 to produce the linear inequalities or equations. All linear inequalities or equations for each operations in ARX ciphers compose the **constraints** of MILP model for linear approximations of ARX ciphers. We can set the **objective function** for r-round linear approximation as the

$\sum_{j=1}^{r}\sum_{i=1}^{n-1} s_i$ to find the minimal value of it. It means to find the optimized linear approximation. We use the Gurobi optimizer to solve the system of inequalities to search for the linear approximations.

As describe in the model of searching for differential characteristics, we assume that the two inputs to modular addition and the consecutive rounds are independent. However, for some ARX constructions, they are not independent. So the practical correlation of our identified linear approximations for some fixed key may vary significantly from that derived from our model.

5 Application to Speck

5.1 Description of Speck

Speck is a family of ARX-based block ciphers proposed by the National Security Agency of the USA in [2], which contains 10 variants. The variants are characterized by the block size of $2n$ bits (where n is the word size) and the key size of mn bits. The Speck block cipher variant with block size $2n$ and key size mn is denoted as Speck$2n/mn$. The rotation constants α, β used in round functions and the number of rounds r are listed in Table 3 for all variants of the Speck.

The round function of Speck consists of XOR, modular addition in $\mathbb{F}_2^n$ and rotation operations. If we denote the subkey in the i-th round as k_i, the input and output of the i-th round as (x_{i-1}, y_{i-1}) and (x_i, y_i), the round function is operated as follows,

$$x_i = ((x_{i-1} \ggg \alpha) \boxplus y_{i-1}) \oplus k_i, \qquad y_i = (y_{i-1} \lll \beta) \oplus x_i,$$

where α and β are rotation constants listed in Table 3.

Table 3: Parameters for Speck Family of Block Ciphers

Variant $2n/mn$	Block Size $2n$	Word Size n	Key Size mn	Key Words m	Rounds r	α	β
32/64	32	16	64	4	22	7	2
48/72	48	24	72	3	22	8	3
48/96	48	24	96	4	23	8	3
64/96	64	32	96	3	26	8	3
64/128	64	32	128	4	27	8	3
96/96	96	48	96	2	28	8	3
96/144	96	48	144	3	29	8	3
128/128	128	64	128	2	32	8	3
128/192	128	64	192	3	33	8	3
128/256	128	64	256	4	34	8	3

5.2 Differential Characteristics and Linear Approximations of Speck

In this subsection, we will give the details how to use the models in Section 3 and Section 4 to search for the differential characteristics and linear approximations for Speck.

Firstly, we produce the system of inequalities to construct the model for the differential or linear trails for Speck based on the methods in Section 3 and Section 4. Then we use Gurobi optimizer to solve our MILP model as [21–24]. Indeed, other MILP optimizers, such as CPLEX [8], can also be used. The procedure of our method is outlined as follows.

Step 1: Convert the system of inequalities describing r rounds of Speck into a format that is readable by Gurobi.

Step 2: Use Gurobi to search for the trails with the input from *Step* 1.

Without loss of generality, we describe how to construct the model to search for the differential characteristic of r-round Speck32. We denote the input difference and the output difference for the i-th round as Δz_{i-1} and Δz_i, respectively, $\Delta z_i = (\Delta z_i^{31}, \ldots, \Delta z_i^0), 0 < i \leq r$.

If we denote the two input differences of modular addition in the i-th round as α_i and β_i and the output difference as γ_i, then we have $\alpha_i = (\Delta z_{i-1}^{22}, \ldots, \Delta z_{i-1}^{16}, \Delta z_{i-1}^{31}, \ldots, \Delta z_{i-1}^{23}), \beta_i = (\Delta z_{i-1}^{15}, \ldots, \Delta z_{i-1}^{0}), \gamma_i = (\Delta z_i^{31}, \ldots, \Delta z_i^{16})$. According to Section 3, we can produce $13 \times (16-1)+5 = 200$ linear inequalities to represent the differential property of $(\alpha_i, \beta_i \rightarrow \gamma_i)$ for modular addition in the i-th round.

For the XOR operation of two branches in the i-th round, the two input differences are $(\Delta z_{i-1}^{13}, \ldots, \Delta z_{i-1}^{0}, \Delta z_{i-1}^{15}, \Delta z_{i-1}^{14})$ and $(\Delta z_i^{31}, \ldots, \Delta z_i^{16})$, and the output difference is $(\Delta z_i^{15}, \ldots, \Delta z_i^0)$. Thus we have

$$\begin{aligned} \Delta z_{i-1}^{13} \oplus \Delta z_i^{31} &= \Delta z_i^{15}, \\ &\vdots \\ \Delta z_{i-1}^{14} \oplus \Delta z_i^{16} &= \Delta z_i^{0}. \end{aligned}$$

Then we use Inequalities (4) to describe the differential property of XOR operation of two branches in the i-th round, so $5 \times 16 = 80$ linear inequalities are produced. Therefore, we use the above produced $200 + 80 = 280$ linear inequalities to describe the differential property of $(\Delta z_{i-1} \rightarrow \Delta z_i)$.

In the similar way, $280 \cdot r$ linear inequalities are derived to describe the differential property of r rounds $(\Delta z_0 \rightarrow \Delta z_1 \rightarrow \ldots \rightarrow \Delta z_r)$. Moreover, one additional condition $\sum_{j=0}^{31} \Delta z_0^j > 0$ is required to ensure the non-zero plaintext difference.

We set the objective function as the minimal value of $\sum_{i=1}^{r} \sum_{j=0}^{32-2} \neg eq(\alpha_i[j], \beta_i[j], \gamma_i[j])$ and convert the above $280 \cdot r + 1$ linear inequalities as constraints into the LP format that is readable by Gurobi. Finally, we use the Gurobi to find the the differential characteristic of r rounds of Speck32.

Similarly, the process of constructing the model to search for the linear approximations for Speck can be implemented using the model in Section 4. Here we will not provide the details about it due to the limited space. The source code is published in https://github.com/fukai6/milp_speck.git.

As we search for the differential and linear trails for Speck with block size greater than 48, we use the splicing heuristic in order to speed up the search process. The splicing heuristic is to search two short trails and concatenate them to produce a long trail. For example, we can search r_1-round differential characteristic with output difference δ and r_2-round differential characteristic with input difference δ to construct an (r_1+r_2)-round differential characteristic. Based on the observation from our identified differential characteristic for small number of rounds and differential characteristics presented in [4], we find that the differential probability probably is better when the left of δ is $0x80$ and the right of δ is 0. In this way, we manually choose different values of r_1 and r_2, and set $\delta = 0x80||0$ as the output difference or input difference to search two differential trails, then concatenate them to produce an $(r_1 + r_2)$-round differential characteristic. For the linear approximation, we set the input mask or output mask as $0x1||0$.

Finally, the best differential characteristics and linear approximations we found are listed in Table 4, Table 5, Table 6 and Table 7, where $\sum_{i=1}^{r} \log_2 p_i$ and $\sum_{i=1}^{r} \log_2 c_i$ are the probability of differential characteristic and the correlation of linear approximation, respectively.

Table 4: Differential Characteristics for Speck32, Speck48 and Speck64

	Speck32			Speck48			Speck64		
i	Δ_L	Δ_R	$\log_2 p_i$	Δ_L	Δ_R	$\log_2 p_i$	Δ_L	Δ_R	$\log_2 p_i$
0	0211	0A04		001202	020002		04092400	20040104	
1	2800	0010	-4	000010	100000	-3	20000820	20200001	-6
2	0040	0000	-2	000000	800000	-1	00000009	01000000	-4
3	8000	8000	0	800000	800004	-0	08000000	00000000	-2
4	8100	8102	-1	808004	808020	-2	00080000	00080000	-1
5	8004	840E	-3	8400A0	8001A4	-4	00080800	00480800	-2
6	8532	9508	-8	608DA4	608080	-9	00480008	02084008	-4
7	5002	0420	-7	042003	002400	-11	06080808	164A0848	-7
8	0080	1000	-3	012020	000020	-5	F2400040	40104200	-13
9	1001	5001	-2	200100	200000	-3	00820200	00001202	-8
10				202001	202000	-3	00009000	00000010	-4
11				210020	200021	-4	00000080	00000000	-2
12							80000000	80000000	0
13							80800000	80800004	-1
14							80008004	84008020	-3
15							808080A0	A08481A4	-5
$\sum_{i=1}^{r} \log_2 p_i$			-30			-45			-62

Note that the differential characteristics in Table 5 have been produced with $r_1 = 11, r_2 = 4$ for Speck64 and $r_1 = 11, r_2 = 5$ for Speck 96, respectively. For Speck128, with the splicing heuristic, we can only get an 18-round differential characteristic with the probability 2^{-126} by setting $r_1 = r_2 = 9$ with reasonable time. Thus, in order to find a better trail, we firstly search for a 13-round differential trail with the splicing heuristic by setting $r_1 = 9$ and $r_2 = 4$, then extend six rounds before the 13-round differential trial to get the 19-round

Table 5: Differential Characteristic for Speck96 and Speck128

i	Δ_L	Δ_R	$\log_2 p_i$	Δ_L	Δ_R	$\log_2 p_i$
0	240004000009	010420040000		0124000400000000	0801042004000000	
1	082020000000	000120200000	-6	0800202000000000	4808012020000000	-6
2	000900000000	000001000000	-4	4800010000000000	0840080100000002	-6
3	000008000000	000000000000	-2	0808080000000006	4A08480800000016	-7
4	000000080000	000000080000	-1	4000400000000032	1042004000000080	-12
5	000000080800	000000480800	-2	0202000000000080	8012020000000480	-7
6	000000480008	000002084008	-4	0010000000000480	0080100000002084	-5
7	0800FE080808	0800EE4A0848	-12	8080000000002080	84808000000124A0	-5
8	000772400040	400000104200	-21	0400000000012440	2004000000080144	-9
9	000000820200	000000001202	-11	2000000000080220	2020000000480801	-9
10	000000009000	000000000010	-4	0000000000480001	0100000002084008	-7
11	000000000080	000000000000	-2	000000000E080808	080000001E4A0848	-8
12	800000000000	800000000000	0	00000000F2400040	4000000000104200	-15
13	808000000000	808000000004	-1	0000000000820200	0000000000001202	-8
14	800080000004	840080000020	-3	0000000000009000	0000000000000010	-4
15	808080800020	A08480800124	-5	0000000000000080	0000000000000000	-2
16	800400008124	842004008801	-9	8000000000000000	8000000000000000	0
17				8080000000000000	8080000000000004	-1
18				8000800000000004	8400800000000020	-3
19				8080808000000020	A084808000000124	-5
$\sum_{i=1}^{r} \log_2 p_i$			-87			-119

differential characteristic in Table 5, where the 6-round trail has been also found with Gurobi.

The linear approximations in Table 6 and Table 7 have been identified with the parameters: $r_1 = 10, r_2 = 3$ for Speck64, $r_1 = 3, r_2 = 12$ for Speck96 and $r_1 = 6, r_2 = 10$ for Speck128.

Table 6: Linear Approximations of Speck32, Speck48 and Speck64

	Speck32			Speck48			Speck64		
i	Γ_L	Γ_R	$\log_2 c_i$	Γ_L	Γ_R	$\log_2 c_i$	Γ_L	Γ_R	$\log_2 c_i$
0	0380	5224		000131	050021		18600010	10724800	
1	4880	4885	-1	018100	200101	-2	1B000000	03104000	-3
2	20A0	2071	-2	000100	000001	-1	18000000	18120000	-2
3	40A0	00C1	-2	000001	000000	0	C0000000	C0100000	-1
4	0080	4001	-3	0D0000	0C0000	-1	04000006	04800006	-1
5	0000	0001	0	606100	606C00	-2	00260030	04200030	-2
6	0004	0004	0	00024A	00620B	-2	01010501	21013181	-5
7	3810	3010	-1	181040	731042	-4	01800126	00018021	-4
8	2180	01C0	-3	D812C0	9802D0	-3	00018100	20000101	-5
9	066A	0608	-2	040600	C4961A	-5	00000100	00000001	-1
10				2484F2	2480F6	-2	00000001	00000000	0
11							09800000	08000000	-1
12							40610000	40680000	-2
13							00024982	00420802	-3
$\sum_{i=1}^{r} \log_2 c_i$		2^{-14}			2^{-22}			2^{-30}	

For the runtime of the searching algorithm, we spent about several hours on personal computer (4 Core, Intel(R) Core(TM) i5 CPU 650 @3.20GHz) for Speck32 and about one day on IBM server (64 Core, Intel(R)Xeon(R) CPU

Table 7: Linear Approximation of Speck96 and Speck128

i	Γ_L	Γ_R	$\log_2 c_i$	Γ_L	Γ_R	$\log_2 c_i$
0	000001000130	040000010021		0001010000018798	6A800101300601C1	
1	000000018100	200000000101	-2	0000018000300720	9400000180300625	-7
2	000000000100	000000000001	-1	0000000181818100	2000000000181B105	-4
3	000000000001	000000000000	0	0000000001800120	0000000000018021	-3
4	0D0000000000	0C0000000000	-1	0000000000018100	2000000000000101	-4
5	604500000000	604C00000000	-2	0000000000000100	0000000000000001	-1
6	00224D000003	006228000003	-4	0000000000000001	0000000000000000	0
7	181070001018	1B105A680018	-12	0980000000000000	0800000000000000	-1
8	001200000010	180210400000	-6	4045000000000000	4048000000000000	-2
9	101000000000	00101A000000	-3	00224D0000000002	0042280000000002	-4
10	001800000000	000010000000	-2	1810600000000010	1A10536C00000010	-9
11	000010000000	000000000000	-1	0012000000000080	1002186000000080	-4
12	000000D00000	000000C00000	-1	0010000000000406	8010130000000406	-3
13	000006041800	000006048000	-2	3680000000002000	3080180000002004	-3
14	000030003490	000030043080	-3	8500000000010181	8524C000000101A1	-4
15	180181910500	800181A10526	-5	8002000000080001	2106000000080100	-6
16				01301A0000404401	0030180000404801	-3
$\sum_{i=1}^{r} \log_2 c_i$		2^{-45}			2^{-58}	

E7330, 2.40GHz) for other variants of Speck. Note that we have searched for all the differential characteristics and linear approximations for Speck32, however, for other variants we aim to only find better trails than the previous ones but we cannot guarantee they are the best trails.

A summary of the differential characteristics and the linear approximations for Speck is provided in Table 2, which shows that we got better differential characteristics and linear approximations for Speck48, Speck64, Speck96 and Speck128.

In order to check the effect of the assumptions of independent inputs to the modular addition and independent rounds for Speck, we experimentally calculate the probability for our identified differential characteristics in Table 4, **??**, 5. As it is not feasible to do this for many rounds, we break down the differential characteristics to small overlapping segments according to the differential probability of the segments. The calculated probability of each one of these segments has been verified experimentally by encrypting sufficiently many random plaintext pairs for some arbitrary keys. The test results are shown in Table 8. In Table 8, the first column is the tested cipher, the second column shows rounds covered by the segment of differential characteristic, the third column is the theoretic differential probability of the corresponding segment of differential characteristic, the fourth column is the total number of random chosen plaintext pairs used in the test, the fifth column is the total number of tested key values, and the last column is the number of keys which can get the calculated differential probability no less than the theoretic differential probability. Note that we only test the last segment from round 12 to round 19 for Speck-128 because the theoretic differential probabilities for the previous segments are too low to be tested. From Table 8, we can see that the number of good keys significantly deviates from the mean for some cases, which is due to

the independent assumptions for Speck cipher. Such deviation will have effect on the success rate of the attacks in the practitioner's perspective.

5.3 Key Recovery Attack on Speck

In [9] Dinur presented a generic key recovery framework for Speck which can extend the differential attack for more rounds. The idea of the framework uses the guess-and-determine technique instead of counting technique for standard key recovery attack. Furthermore, the cryptanalytic technique of ARX cipher is utilized to speed up the attack on Speck.

For Speck$2n/mn$, if the differential characteristic with probability p for $r-1$ rounds has been found, the attacker can add one round at the top of the differential characteristic and m rounds at the bottom of the differential characteristic to cover $r+m$ rounds in total. It is not necessary to guess the subkey in the first round as it has no effect on the input difference. At last, the attacker can recover the mn-bit secret key of a variant with $r+m$ rounds of Speck$2n/mn$ using $2 \cdot p^{-1}$ chosen plaintexts with time complexity of $2 \cdot p^{-1} \cdot 2^{(m-2)n}$ and memory complexity of 2^{22} bytes.

With our identified differential characteristics for Speck, we can give the improved key recovery attack. Since the attack is same as that in [9], details are omitted. For each variant of Speck, we summarize our attacks and the previous differential attacks in Table 1, which shows that our attacks for the variants of Speck with block size 48, 64, 96 and 128 are best attacks in terms of the number of rounds.

Table 8: Test for Differential Characteristics in Table 4 and 5

Cipher	Rounds of Segment	Differential Probability	Number of Plaintext Pairs	Total Number of Keys	Number of Good Keys
Speck32/64	0-9	2^{-30}	2^{32}	7040	3456
Speck48/96	0-6	2^{-19}	2^{22}	10000	4093
Speck48/96	6-11	2^{-26}	2^{30}	6400	2989
Speck64/128	0-6	2^{-19}	2^{22}	10000	5513
Speck64/128	6-8	2^{-20}	2^{24}	10000	4887
Speck64/128	8-15	2^{-23}	2^{26}	10000	4918
Spec96/144	0-5	2^{-15}	2^{18}	10000	5123
Spec96/144	5-7	2^{-16}	2^{20}	10000	5039
Spec96/144	7-8	2^{-21}	2^{24}	10000	5454
Spec96/144	8-11	2^{-17}	2^{20}	10000	5020
Spec96/144	11-16	2^{-18}	2^{22}	10000	4645
Spec128/256	12-19	2^{-23}	2^{26}	10000	4876

6 Conclusion

In this paper, we construct the MILP model to automatically search for differential and linear approximations for ARX ciphers by researching the differential and linear property of modular addition under the assumptions of independent inputs to the modular addition and independent rounds. Then we use the new MILP model to search for the differential characteristics and linear approximations for Speck cipher. Compared with the previous best differential characteristics for them, our identified differential characteristics for Speck64, Speck96 and Speck128 are extended for one, three and five rounds, respectively, and our differential characteristic for Speck48 has higher probability. We use those new differential characteristics to improve the currently best public attacks on the four variants of Speck. In addition, we searched for the linear approximations for Speck cipher and improved the previous linear approximations for Speck variants with block size greater than 32.

Acknowledgments This work has been supported by 973 Program (No. 2013C B834205), NSFC Projects (No. 61133013, No. 61572293), Program for New Century Excellent Talents in University of China (NCET- 13-0350).

References

1. Abed, F., List, E., Lucks, S., Wenzel, J.: Differential Cryptanalysis of Round-Reduced SIMON and SPECK. FSE 2014. LNCS, vol. 8540, pp. 525-545. Springer, 2015.
2. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The SIMON and SPECK Famillies of Lightweight Block Ciphers. Cryptology ePrint archive, Report 2013/543 (2013). http://eprint.iacr.org/
3. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. Journal of Cryptology, 4(1):3-72, 1991.
4. Biryukov, A., Velichkov, V.: Automatic Search for Differential Trails in ARX Ciphers. CT-RSA 2014. LNCS, vol. 8366, pp. 227-250. Springer, 2014.
5. Biryukov, A., Velichkov, V., Corre, Y.: Automatic Search for the Best Trails in ARX: Application to Block Cipher Speck. FSE 2016, Bochum, Germany, March 20–23 (to appear), 2016.
6. Biryukov, A., Roy, A., Velichkov, V.: Differential Analysis of Block Ciphers SIMON and SPECK. FSE 2014. LNCS, vol. 8540, pp. 546-570. Springer, 2015.
7. Biryukov, A., Velichkov, V., Corre, Y. L.: Automatic Search for the Best Trails in ARX: Application to Block Cipher Speck. FSE 2016. LNCS. Springer, to Appear.
8. CPLEX, I. I.: IBM software group. User-Manual CPLEX. 2011.
9. Dinur, I.: Improved Differential Cryptanalysis of Round-Reduced Speck. SAC 2014. LNCS, vol. 8781, pp. 147-164. Springer, 2014.
10. De Cannière, C., Rechberger, C.: Finding SHA-1 Characteristics: General Results and Applications. ASIACRYPT 2006. LNCS, vol. 4284, pp. 1–20. Springer, 2006.
11. Gurobi Optimization. Gurobi Optimizer Reference Manual. 2013. http://www.gurobi.com

12. Leurent, G.: Construction of Differential Characteristics in ARX Designs-Application to Skein. Cryptology ePrint Archive, 2012/668(2012). https://eprint.iacr.org/
13. Lipmaa, H., Moriai, S.: Efficient Algorithms for Computing Differential Properties of Addition. FSE 2002. LNCS, vol. 2355, pp. 336-350. Springer, 2002.
14. Liu, M., Chen, J.: Improved Linear Attacks on the Chinese Block Cipher Standard. JCST, vol. 29, Issue 6, pp. 1123-1133. Springer, 2014.
15. Matsui, M.: Linear Cryptanalysis Method for DES Cipher. EUROCRYPT 1993. LNCS, vol.765, pp. 386-397. Springer, 2001.
16. Matsui, M.: On Correlation between the Order of S-boxes and the Strength of DES. EUROCRYPT 94. LNCS, vol. 950, pp. 366-375. Springer, 2006.
17. Mendel, F., Nad, T., Schläffer, M.: Finding SHA-2 Characteristics: Searching through a Minefield of Contradictions. ASIACRYPT 2011. LNCS, vol. 7073, pp. 288–307. Springer, 2011.
18. Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming. Inscrypt 2011. LNCS, vol. 7537, pp. 57-76. Springer, 2012.
19. Nyberg, K., Wallen, J.: Improved Linear Distinguishers for SNOW 2.0. FSE 2006. LNCS, vol. 4047, pp. 144-162. Springer, 2006.
20. Stein, W.,et al.: Sage: Open Source Mathematical Software(2008).
21. Sun, S., Hu, L., Song, L., Xie, Y., Wang, P.: Automatic Security Evaluation of Block Ciphers with S-bP Structures Against Related-Key Differential Attacks. Inscrypt 2013. LNCS, vol.8567, pp. 39-51. Springer, 2013.
22. Sun, S., Hu, L., Qiao, K., Ma, X., Song, L.: Automatic Security Evaluation and (Related-key) Differential Characteristic Search: Application to SIMON, PRESENT, LBlock, DES(L) and Other Bit-Oriented Block Ciphers. ASIACRYPT 2014. LNCS, vol. 8873, pp. 158-178. Springer, 2014.
23. Sun, S., Hu, L., Qiao, K., Ma, X., Song, L.: Automatic Security Evaluation and (Related-key) Differential Characteristic Search: Application to SIMON, PRESENT, LBlock, DES(L) and Other Bit-Oriented Block Ciphers. Cryptology ePrint Archive, Report 2013/676 (2013). https://eprint.iacr.org/
24. Sun, S., Hu, L., Wang, M., Wang, P., Qiao, K., Ma, X., Shi, D., Song, L., Fu, K.: Towards Finding the Best Characteristics of Some Bit-oriented Block Ciphers and Automatic Enumeration of (Related-key) Differential and Linear Characteristics with Predefined Properties. Cryptology ePrint Archive, Report 2014/747 (2014). https://eprint.iacr.org/
25. Wallén, J.: Linear Approximations of Addition Modulo 2^n. FSE 2003. LNCS, vol. 2887, pp. 261-273. Springer, 2003.
26. Wang, G., Keller, N., Dunkelman, O.: The Delicate Issues of Addition with Respect to XOR Differences. SAC 2007. LNCS, vol. 4786, pp. 212-231. Springer, 2007.
27. Winnen, L.: Sage S-box Milp Toolkit. http://www.ecrypt.eu.org/tools/sage-s-box-milp-toolkit
28. Wu, S., Wang, M.: Security Evaluation against Differential Cryptanalysis for Block Cipher Structures . Cryptology ePrint Archive, Report 2011/551 (2011). https://eprint.iacr.org/
29. Wu, S., Wu, H., Huang, T., Wang, M., Wu, W.: Leaked-state-forgery attack against the authenticated encryption algorithm ALE. ASIACRYPT 2013. LNCS, vol. 8269, pp. 377-404. Springer, 2013.
30. Yao, Y., Zhang, B., Wu, W.: Automatic Search for Linear Trails of the SPECK Family. ISC 2015. LNCS, vol. 9290, pp. 158–176. Springer, 2015.