New Methods for Bounding the Length of Impossible Differentials of SPN Block Ciphers

Senpeng Wang[1,2], Dengguo Feng[1], Bin Hu[2], Jie Guan[2], Ting Cui[2], Tairong Shi[2], and Kai Zhang[2]

[1] State Key Laboratory of Cryptology, Beijing, China, wsp2110@126.com
[2] PLA SSF Information Engineering University, Zhengzhou, China

Abstract. Impossible differential (ID) cryptanalysis is one of the most important cryptanalytic approaches for block ciphers. How to evaluate the security of Substitution-Permutation Network (SPN) block ciphers against ID is a valuable problem. In this paper, a series of methods for bounding the length of IDs of SPN block ciphers are proposed. From the perspective of overall structure, we propose a general framework and three implementation strategies. The three implementation strategies are compared and analyzed in terms of efficiency and accuracy. From the perspective of implementation technologies, we give the methods for determining representative set, partition table and ladder and integrating them into searching models. Moreover, the rotation-equivalence ID sets of ciphers are explored to reduce the number of models need to be considered. Thus, the ID bounds of SPN block ciphers can be effectively evaluated. As applications, we show that 9-round PRESENT, 8-round GIFT-64, 12-round GIFT-128, 5-round AES, 6-round Rijndael-160, 7-round Rijndael-192, 7-round Rijndael-224, 7-round Rijndael-256 and 10-round Midori64 do not have any ID under the sole assumption that the round keys are uniformly random. The results of PRESENT, GIFT-128, Rijndael-160, Rijndael-192, Rijndael-224, Rijndael-256 and Midori64 are obtained for the first time. Moreover, the ID bounds of AES, Rijndael-160, Rijndael-192, Rijndael-224 and Rijndael-256 are infimum.

Keywords: Impossible differential · PRESENT · GIFT · Midori64 · Rijndael · AES

1 Introduction

Impossible differential (ID) cryptanalysis [Knu98,BBS99] is one of the most effective cryptanalytic approaches for block ciphers. The main idea of it is to utilize IDs (differentials with probability 0) to discard wrong keys. So far, ID cryptanalysis has been used to attack lots of block ciphers, such as AES [MDRM10].

For attackers, finding ID plays an important role in ID attack. In [KHS$^+$03], Kim *et al.* proposed the first automatic method for finding IDs, called $\mathcal{U}$-method. After that, many improved automatic tools are presented, such as UID-method [LLWG14], $\mathcal{WW}$-method [WW12], $\mathcal{U}^\star$-method [SGWW20], *etc.* However, all these tools treat S-boxes as ideal ones that any nonzero input difference could produce every nonzero output difference. Thus, the IDs obtained by these methods may not be the longest for real ciphers. In order to tackle this problem, Cui *et al.* [CJF+16] and Sasaki and Todo [ST17b] independently proposed automatic tools based on Mixed Integer Linear Programming (MILP) to search IDs for block ciphers with the differential details of S-box considered. With the tools based on MILP, they can identify whether a specific differential is ID. In theory, the tools based on MILP can find all IDs under the assumption that round keys are uniformly random. However, for a block cipher with n-bit block size, the number of differentials in the whole search space is about 2^{2n} which is not affordable to determine all these differentials one by one.

For designers, it is important to evaluate the security of block ciphers. To prove the security of a block cipher against ID attacks, a common way is to give an upper bound on the rounds of ID. In [CJZ+17], Cui *et al.* suggested that the differential pattern matrix of the P-layer could be used to deduce all IDs for SPN block ciphers. At EUROCRYPT 2016, Sun *et al.* [SLG+16] associated a primitive index with the characteristic matrix of the linear layer. They proved that the length of ID for some special SPN block ciphers was bounded by the primitive index of the linear layer. In order to obtain the bounds of ID in practical time, they proved that under special conditions whether there existed ID depended only on the existence of low-weight ID. To overcome the limitations of the above methods, Wang and Jin [WJ21] used linear algebra to propose a practical method that could give the upper bound on the length of ID for any SPN block cipher when treating S-boxes as ideal ones. Since the above methods do not consider the differential details of S-box, their bounds may become invalid.

When the details of S-box are considered, the security bounds of ciphers against ID will be more convincing. The difficulty of this problem is that it needs to prove that all differentials are possible when the round number of a block cipher is not less than a certain integer. If there is no special explanation, all the contents of ID considering the details of the S-box in this paper are obtained under the assumption that round keys are uniformly random. The research progress in this field can be divided into the following three categories.

- **Rigorous mathematical derivation.** By revealing some important properties of the S-box and linear layer used in AES, Wang and Jin [WJ19] prove that even though the details of the S-box are considered, there do not exist ID covering more than 4 rounds for AES. However, this method is only applicable to AES at present.
- **Bounds on partial search space.** The automatic search methods based on solvers [CJF+16,ST17b,BC20] can determine whether a concrete differential is ID. Thus, the bound on partial search space of differentials can be obtained.
- **Bounds on whole search space for special SPN ciphers.** At SAC 2022, Hu *et al.* [HPW22] partitioned the whole search space of difference pairs into lots of small disjoint sets. When the number of sets is reduced to a reasonable size, they can detect whether there exist ID with MILP models. Due to the

limitation of huge time complexity, their method currently works only for special SPN cipher whose block size is 64 bits.

1.1 Our Contributions

In this paper, we propose a series of methods for bounding the length of IDs of SPN block ciphers. The contributions can be classified into three parts.

- **A general framework and three implementation strategies.** Based on our new definition about the set of difference pairs, called *ladder* (a set whose every input difference can propagate to every output difference), we propose a general framework for bounding the length of IDs of SPN block ciphers. The framework divides the whole cipher into small components and constructs a ladder for a middle component. Thus, the input and output differences can be considered separately. Then, three implementation strategies of the framework are introduced. We compare and analyze the three implementation strategies in terms of efficiency and accuracy. Thus, we can choose appropriate strategy according to specific block ciphers.
- **More efficient and accurate implementation technologies.** In order to reduce the implementation complexity, we put forward the definitions of *optimal representative set* and *optimal partition table*. For small-size S-box (e.g. 4-bit or 8-bit) and middle-size S-box (e.g. 16-bit), we give corresponding algorithms to determine the optimal representative set and partition table. For large-size superbox (e.g. 32-bit), a heuristic algorithm is proposed to determine a relatively good representative set and partition table. Thus, compared with the work in [HPW22], our methods can use fewer or even the least models to obtain the security evaluation against ID.

  In addition, we propose the definition of *maximal ladder* to guide the selection of a better ladder. Then, the methods for determining a maximal ladder of S-box layer and integrating it into searching model are given. Moreover, the rotation-equivalent ID sets of ciphers are explored to reduce the number of models need to be considered. Thus, we can bound the length of IDs of SPN block ciphers effectively.
- **Applications to SPN block ciphers.** Under the sole assumption that round keys are uniformly random, we show that 9-round PRESENT, 8-round GIFT-64, 12-round GIFT-128, 5-round AES, 6-round Rijndael-160, 7-round Rijndael-192, 7-round Rijndael-224, 7-round Rijndael-256 and 10-round Midori64 do not have any ID. The results of PRESENT, GIFT-128, Rijndael-160, Rijndael-192, Rijndael-224, Rijndael-256 and Midori64 are obtained for the first time. Moreover, the ID bounds of AES, Rijndael-160, Rijndael-192, Rijndael-224 and Rijndael-256 are infimum.

Compared with the methods in [HPW22], our methods have two advantages. On one hand, our methods are more general which are no longer limited to special SPN ciphers with 64-bit block size. For instance, under the sole assumption that round keys are uniformly random, the ID bound of GIFT128 is obtained for the first time. On the other hand, our methods are more efficient. For example, when determining whether there is ID for 8-round GIFT-64, the methods in [HPW22] need to solve 2^{26} fundamental models, while our methods only need to solve $2^{24.68}$ fundamental models. All the application results are shown in Table 1.

Table 1. The ID results of some SPN block ciphers

Cipher	Block size	Longest known ID	Number of models	Bound	Reference
PRESENT	64	6 [HLJ$^+$20]	-	$7^\star$	[HLJ$^+$20]
			$2^{24.68}$	9	Sect. 5.1
GIFT-64	64	6 [HLJ$^+$20]	-	$7^\star$	[BPP$^+$17]
			2^{26}	8	[HPW22]
			$2^{24.68}$	8	Sect. 5.2
GIFT-128	128	7 [HPW22]	$2^{12.17}$	8^*	[HPW22]
			$2^{25.83}$	12	Sect. 5.2
AES (Rijndael-128)	128	4 [MDRM10]	-	5	[WJ19]
			$75 + \mathcal{O}\left(2^{32}\right)^\blacklozenge$	5	Sect. 6.1
Rijndael-160	160	5 [ZWP$^+$08]	217	6	Sect. 6.1
Rijndael-192	192	6 [JP07]	-	$7^\dagger$	[HPW22]
			819	7	Sect. 6.1
Rijndael-224	224	6 [JP07]	2413	7	Sect. 6.1
Rijndael-256	256	6 [ZWP$^+$08]	8925	7	Sect. 6.1
Midori64	64	5 [BBI$^+$15]	-	$6^\star$	[BBI$^+$15]
			2^{24}	10	Sect. 6.2

* The security bound of the search space where there is only one active S-box for both the input and output differences.
* The security bound of the search space where there is only one active superbox for both the input and output differences.
† The security bound of truncated ID omitting the details of S-box.
$^\blacklozenge$ We need to verify some representatives of 32-bit superboxes in AES.

1.2 Outline

This paper is organized as follows: Sect. 2 introduces the notations, definitions and related works. In Sect. 3, we propose a general framework and three implementation strategies for bounding the length of IDs. In Sect. 4, the implementation technologies are detailed. In Sect. 5 and 6, we apply our methods to two types of SPN block ciphers. In Sect. 7, we conclude the paper.

2 Preliminaries

2.1 Notations and Definitions

Some notations used in this paper are defined in Table 2.

Table 2. Some notations used in this paper

$\mathbb{F}_2$	The finite field $\{0,1\}$
$x \in \mathbb{F}_2^n$	An n-bit vector or difference
$x \oplus y$	Bitwise XOR of x and y
$x \lll i$	Left rotation of x by i-bit position
$x \ggg i$	Right rotation of x by i-bit position
$x\|\|y$	The concatenation of x and y
$x^{n\|\|}$	The concatenation $x\|\|x\|\|\cdots\|\|x$ whose number of x is n
$\emptyset$	Empty set
A	Set is denoted as uppercase letter such as A
$\|A\|$	The number of elements in the set A
$A \cap B$	The intersection of two sets A and B
$A \cup B$	The union of two sets A and B
$A + B$	If $A \cap B = \emptyset$, we denote the union of A and B as $A + B$
$A - B$	The set $\{a \mid a \in A \text{ and } a \notin B\}$
$A \otimes B$	The set $\{(a,b) \mid a \in A, b \in B\}$
A^n	The set $A \otimes A \otimes \cdots \otimes A$ whose number of A is n

Definition 1. (Expected Differential Probability [CR15]). *Let $f_k : \mathbb{F}_2^n \times \mathbb{F}_2^\kappa \to \mathbb{F}_2^m$ be a keyed vectorial boolean function with κ-bit key size. Then, the expected probability of differential $(a,b) \in \mathbb{F}_2^n \times \mathbb{F}_2^m$ over f_k is defined as:*

$$EDP(a \xmapsto{f_k} b) = 2^{-\kappa} \sum_{k \in \mathbb{F}_2^\kappa} DP(a \xmapsto{f_k} b),$$

where $DP(a \xmapsto{f_k} b) = 2^{-n} \times |\{x \in \mathbb{F}_2^n | f_k(x) \oplus f_k(x \oplus a) = b\}|$ is the differential probability of (a,b) over f_k.

If $EDP(a \xmapsto{f_k} b) = 0$, the differential (a,b) is an ID over f_k, denoted as $a \overset{f_k}{\nrightarrow} b$. Otherwise, if $EDP(a \xmapsto{f_k} b) > 0$, the differential (a,b) is a possible differential pattern, denoted as $a \overset{f_k}{\rightarrow} b$. For two sets of differences A and B, if $a \overset{f_k}{\rightarrow} b$ holds for all $(a,b) \in A \otimes B$, we denote it as $A \overset{f_k}{\rightarrow} B$. Otherwise we denote it as $A \overset{f_k}{\nrightarrow} B$. Moreover, $a \overset{f_k}{\rightarrow} B$ and $A \overset{f_k}{\rightarrow} b$ are equivalent to $\{a\} \overset{f_k}{\rightarrow} B$ and $A \overset{f_k}{\rightarrow} \{b\}$, respectively.

In this paper, we are only interested in the bit-wise XOR difference. On this condition, we introduce the following definition and theorem.

Definition 2. (Markov Cipher [LMM91]). *An iterated cipher with round function $f_k(x) = f(x \oplus k)$ is a Markov cipher, if for all choices of a and b $(a \neq 0, b \neq 0)$, the probability*

$$P(f_k(x) \oplus f_k(x') = b | x \oplus x' = a, x = c)$$

is independent of c when the round key is uniformly random.

Theorem 1. (EDP of Markov Cipher [LMM91]). *Let $E_k = f_{k_{r-1}} \circ f_{k_{r-2}} \circ \cdots \circ f_{k_0}$ be an r-round Markov cipher, where k_i is the round key and $f_{k_i}(x) = f(x \oplus k_i)$ holds for all $0 \le i \le r-1$. Then, under the assumption that round keys are uniformly random, the EDP of (a_0, a_r) over E_k can be calculated as*

$$EDP(a_0 \xmapsto{E_k} a_r) = \sum_{a_1} \sum_{a_2} \cdots \sum_{a_{r-1}} EDP(a_0 \xmapsto{f_{k_0}} a_1 \xmapsto{f_{k_1}} \cdots \xmapsto{f_{k_{r-1}}} a_r), \tag{1}$$

where $EDP(a_0 \xmapsto{f_{k_0}} a_1 \xmapsto{f_{k_1}} \cdots \xmapsto{f_{k_{r-1}}} a_r) = \prod_{i=0}^{r-1} EDP(a_i \xmapsto{f_{k_i}} a_{i+1})$ is the EDP of the r-round differential trail $a_0 \longmapsto a_1 \longmapsto \cdots \longmapsto a_r$ over E_k.

According to Eq. (1), for an r-round Markov cipher E_k, if we want to prove $a_0 \overset{E_k}{\to} a_r$, we need to find an r-round possible differential trail satisfying $EDP(a_0 \xmapsto{f_{k_0}} a_1 \xmapsto{f_{k_1}} \cdots \xmapsto{f_{k_{r-1}}} a_r) > 0$. If we want to prove that there does not exist any ID for cipher E_k, we have to prove that $a_0 \overset{E_k}{\to} a_r$ holds for every concrete differential (a_0, a_r). As far as we know, almost all SPN block ciphers (such as AES [DR02]) are Markov ciphers. For those SPN ciphers that are not Markov ciphers (such as SKINNY [BJK$^+$16]), we should not misuse the result of Theorem 1.

2.2 Current Automatic Methods for Finding IDs

In [MWGP11,SHW$^+$14], MILP based methods for searching differential distinguishers were proposed. By adding additional constraints on the input and output differences, Cui *et al.* [CJF$^+$16] and Sasaki and Todo [ST17b] independently proposed MILP models to search IDs for block ciphers with the details of S-box considered. Using MILP tools, they are able to identify whether a differential is ID or not. However, when we want to find all the IDs or to know whether there exist longer ID for a block cipher, we have to solve about 2^{2n} models for a cipher with n-bit block size to check all input and output difference pairs. The search space far exceeds the existing computing power.

In order to tackle this problem, Hu *et al.* [HPW22]) partitioned the whole search space into many small disjoint sets and then excluded the sets containing no ID. Thus, when their methods have determined that all differentials are not IDs, the provable security of ciphers against ID can be obtained. We will introduce their methods from the perspective of bounding the length of IDs which is also the main topic of this paper.

Definition 3. (Representative Set [HPW22]). *For a function f, let A and B be the sets of input and output differences, respectively. If the following condition is satisfied,*

$$\forall a \in A, \exists b \in B \text{ satisfying } a \overset{f}{\to} b$$

we call B the representative set of A over f, denoted as $A \overset{f}{\to} \exists B$.

Definition 4. (Partition Table [HPW22]). *If* $A \xrightarrow{f} \exists B$, *then*

$$\bigcup_{b \in B} \{a \in A | a \xrightarrow{f} b\} = A.$$

For any $a \in A$, *we remove the overlapping elements and make it exist in only one set of* $\{a \in A | a \xrightarrow{f} b\}, b \in B$. *Thus, we get a partition of* A *which can be stored in a hash table* H *with* $b \in B$ *as key and the value* $H[b]$ *is the set* $\{a \in A | a \xrightarrow{f} b\}$ *after removing. Thus,* $A = \sum_{b \in B} H[b]$ *is a partition table, denoted as* $PT[A, B, H, f]$.

However, it is very difficult to determine the representative sets and partition tables of a cipher directly. By dividing a large-dimension function into small parts, Hu *et al.* [HPW22] proposed a solution as follow.

Theorem 2. ([HPW22]). *For a function* S *comprising of* m *parallel S-boxes, denoted as* $S = s_{m-1} || \cdots || s_1 || s_0$, *let* $A = A_{m-1} \otimes \cdots \otimes A_1 \otimes A_0$ *be the input difference set of* S, *where* A_i *is the input difference set of* $s_i, i \in \{0, 1, \ldots, m-1\}$. *If we obtain the partition tables* $PT\left(A_i, B_i, H_i, s_i\right), i \in \{0, 1, \ldots, m-1\}$, *then*

$$A = \sum_{b_{m-1} \in B_{m-1}} \cdots \sum_{b_1 \in B_1} \sum_{b_0 \in B_0} H_{m-1}[b_{m-1}] \otimes \cdots \otimes H_1[b_1] \otimes H_0[b_0]$$

Thus, we obtain the partition table of A *over* S.

Then, Hu *et al.* [HPW22] proposed a framework for bounding the length of IDs as showed in the following theorem (also illustrated in Fig. 1)

Theorem 3. (Bounding the Length of IDs [HPW22]). *For a cipher* $E = E_2 \circ E_1 \circ E_0$ *and partition tables* $PT[A_0, A_1, H_0, E_0]$ *and* $PT[A_3, A_2, H_2, E_2^{-1}]$, *the set* $A_0 \otimes A_3$ *is the union of smaller sets as follows,*

$$A_0 \otimes A_3 = \sum_{a_1 \in A_1, a_2 \in A_2} H_0[a_1] \otimes H_2[a_2].$$

For each element $(a_1, a_2) \in A_1 \otimes A_2$, *the model is built to detect whether* $a_1 \xrightarrow{E_1} a_2$. *If* $A_1 \xrightarrow{E_1} A_2$, *the cipher* E *has no ID over* $A_0 \otimes A_3$. *Thus, the ID bound of* E *can be obtained. Otherwise, if there exists* $a_1 \overset{E_1}{\nrightarrow} a_2$, *the set of difference pairs* $H_0[a_1] \otimes H_2[a_2]$ *may contain some IDs.*

The above framework considers the input difference set and output difference set together. In order to get the ID bound of E, at least $|A_1| \times |A_2|$ models need to be solved. The number of models may not affordable. A natural question is whether we can consider input difference set and output difference set separately. Following this initial idea, we propose a general framework and its implementation strategies in Sect. 3.

3 Overall Structure of Bounding the Length of IDs

In this part, we propose a general framework for bounding the length of IDs. Based on the framework, three implementation strategies are showed.

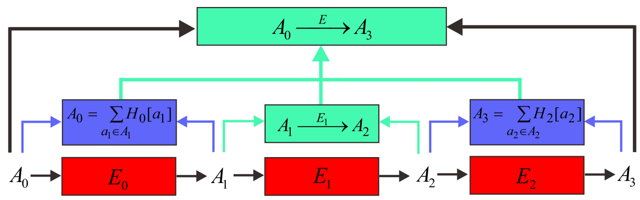


Fig. 1. The framework for bounding the length of IDs in [HPW22]

3.1 A General Framework

Definition 5. (Ladder) *For a function f, let A and B be sets of input and output differences, respectively. If the condition $A \xrightarrow{f} B$ is satisfied, we call $A \otimes B$ the ladder of f.*

Theorem 4. *For a bijective function f, if $A \otimes B$ is a ladder of f, then $B \otimes A$ is also a ladder of f^{-1}, where f^{-1} is the inverse function of f.*

Proof. Because $A \xrightarrow{f} B$, for any $(a, b) \in A \otimes B$, there exists x satisfying $f(x) \oplus f(x \oplus a) = b$. For the element $y = f(x)$, we have $f^{-1}(y) \oplus f^{-1}(y \oplus b) = x \oplus (x \oplus a) = a$. Thus, for any $(b, a) \in B \otimes A$, we have $b \xrightarrow{f^{-1}} a$. □

Based on the definitions of representative set, partition table and ladder, we propose a general framework for bounding the length of IDs as showed in the following theorem (also illustrated in Fig. 2).

Theorem 5. *Let $E = E_4 \circ E_3 \circ E_2 \circ E_1 \circ E_0$ be a cipher, where $E_i, 0 \leq i \leq 4$ are all bijective functions. if there exist the sets of differences $A_0, A_1, A_2, A_3, A_4, A_5$ and partition tables $PT[A_0, A_1, H_0, E_0], PT[A_5, A_4, H_4, E_4^{-1}]$ satisfying*

$$\begin{cases} A_1 \xrightarrow{E_1} \exists A_2, \\ A_2 \xrightarrow{E_2} A_3, \\ A_4 \xrightarrow{E_3^{-1}} \exists A_3, \end{cases} \tag{2}$$

we have $A_0 \overset{E}{\nrightarrow} A_5$. That is, the cipher E has no ID over $A_0 \otimes A_5$.

Proof. Because $PT[A_0, A_1, H_0, E_0]$, we have $A_0 = \sum_{a_1 \in A_1} H_0[a_1]$. For any difference $a_0 \in A_0$, there exists $a_1 \in A_1$ satisfying $a_0 \xrightarrow{E_0} a_1$. According to Definition 3, if $A_1 \xrightarrow{E_1} \exists A_2$, for any $a_1 \in A_1$, there exists $a_2 \in A_2$ satisfying $a_1 \xrightarrow{E_1} a_2$. Therefore, for any difference $a_0 \in A_0$, there exists $a_2 \in A_2$ satisfying

$$a_0 \xrightarrow{E_1 \circ E_0} a_2. \tag{3}$$

Similarly, for any $a_5 \in A_5$, there exists $a_3 \in A_3$ satisfying $a_5 \xrightarrow{E_3^{-1}\circ E_4^{-1}} a_3$. Because $E_3^{-1} \circ E_4^{-1}$ is a bijective function, according to Theorem 4, for any difference $a_5 \in A_5$, there exists $a_3 \in A_3$ satisfying

$$a_3 \xrightarrow{E_4 \circ E_3} a_5. \tag{4}$$

Because $A_2 \xrightarrow{E_2} A_3$, we have

$$a_2 \xrightarrow{E_2} a_3. \tag{5}$$

Combining the Eq. (3), (4) and (5) together, for any $a_0 \in A_0$ and $a_5 \in A_5$, there exist $a_2 \in A_2$ and $a_3 \in A_3$ satisfying

$$a_0 \xrightarrow{E_1 \circ E_0} a_2 \xrightarrow{E_2} a_3 \xrightarrow{E_4 \circ E_3} a_5.$$

Thus, we have $A_0 \xrightarrow{E} A_5$. □

According to Eq. (2), the partition tables of input difference set A_0 and output difference set A_5 can be considered separately. This will improve the efficiency of security evaluation against ID. Moreover, if the functions E_1 and E_3 are identical permutation, the framework degenerates into the method as shown in Theorem 3. Thus, our framework is more general.

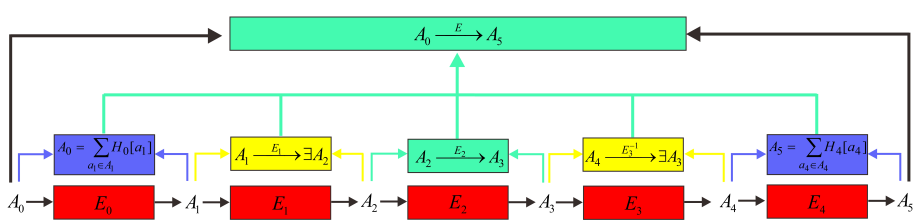


Fig. 2. A general framework for bounding the length of IDs

3.2 Three Implementation Strategies

In this part, three implementation strategies are proposed to bound the length of IDs. To facilitate the description of the strategies, we introduce an indicator variable $flag$ to denote the results of ID as following:

$$flag = \begin{cases} 0, & \text{if there is no ID,} \\ 1, & \text{if there is at least one ID,} \\ 2, & \text{if cannot determine whether there is ID.} \end{cases}$$

When we cannot get the value of $flag$ due to the limited storage and computing capacity, we set $flag = 2$.

3.2.1 Partition First Implementation Strategy This strategy will first obtain the partition tables of the input and output difference sets. Then, if every representative difference of input differences can propagate to every representative difference of output differences, we can obtain the ID bound. This strategy is similar to the method shown in Theorem 3. However, we introduce this strategy from the perspective of ladder. Moreover, when there are some uncertain IDs, we adopt a different enhance stage.

For a cipher $E = E_2 \circ E_1 \circ E_0$, we construct partition tables $PT[A_0, A_1, H_0, E_0]$ and $PT[A_3, A_2, H_2, E_2^{-1}]$, where A_0 and A_3 are the input and output difference sets of E, respectively. In the fundamental stage, if $A_1 \otimes A_2$ is a ladder of E_1, according to Theorem 5, there is no ID for E over $A_0 \otimes A_3$. If $A_1 \otimes A_2$ is not a ladder of E_1, we obtain a set $I = \{(a_1, a_2) \in A_1 \otimes A_2 | a_1 \overset{E_1}{\nrightarrow} a_2\}$. And we need to further determine whether $H_0[a_1] \otimes H_2[a_2], (a_1, a_2) \in I$ are ladders of E. In the enhance stage, we construct a set $I_1 = \{a_1 \in A_1 | \, (a_1, a_2) \notin I$ holds for every $a_2 \in A_2\}$. Because for any $a_1 \in I_1$, we have $a_1 \overset{E_1}{\rightarrow} A_2$. Thus, $\sum_{a_1 \in I_1} H_0[a_1] \overset{E}{\rightarrow} A_3$. Therefore, for any $a_1 \in A_1$, we can reduce the hash table $H_0[a_1]$ to $H_0'[a_1] = H_0[a_1] - \sum_{a \in I_1} H_0[a]$. Similarly, for any $a_2 \in A_2$, we can obtain the reduced hash table $H_2'[a_2]$. Then, for any $(a_1, a_2) \in I$, we further explore whether $H_0'[a_1] \overset{E}{\rightarrow} H_2'[a_2]$. The whole procedure for obtaining the ID result of E over $A_0 \otimes A_3$ is demonstrated in Algorithm 1.

From **Line 3** in Algorithm 1, we know that $|A_1| \times |A_2|$ models need to be build to obtain ID result of E. The partition tables $PT[A_0, A_1, H_0, E_0]$ and $PT[A_3, A_2, H_2, E_2^{-1}]$ will have an important influence on the time complexity of Algorithm 1. In [HPW22], Hu *et al.* proposed an intuitive algorithm which could generate representative sets and partition tables. Just as they write in the paper, their algorithm is not very efficient. On one hand, their method cannot be applied into large-size S-box (e.g. 32-bit). On the other hand, their method cannot guarantee the obtained representative sets and partition tables are optimal representative sets and partition tables. Thus, we propose the definitions of optimal representative set and partition table in Sect. 4.1. Compared with the methods proposed in [HPW22], our methods can use fewer or even least models to obtain the ID bound.

3.2.2 Ladder First Implementation Strategy Different from partition first implementation strategy, ladder first implementation strategy directly construct a ladder to separate the input difference set and output difference set. Thus, we can obtain the ID result by independently researching the input difference set and output difference set. This divide and conquer method will greatly reduce the number of models need to be solved.

For a cipher $E = E_4 \circ E_3 \circ E_2 \circ E_1 \circ E_0$, we construct a ladder $A_2 \overset{E_2}{\rightarrow} A_3$ and two partition tables $PT[A_0, A_1, H_0, E_0]$ and $PT[A_5, A_4, H_4, E_4^{-1}]$, where A_0 and A_5 are the input and output difference sets of E, respectively. In the fundamental stage, if $A_1 \overset{E_1}{\rightarrow} \exists A_2$ and $A_4 \overset{E_3^{-1}}{\rightarrow} \exists A_3$, according to Theorem 5, there is no ID for E over $A_0 \otimes A_5$. Otherwise, we obtain two sets $I = \{a_1 \in A_1 | a_1 \overset{E_1}{\nrightarrow} \exists A_2\}$

Algorithm 1 Partition first implementation strategy

Input: The cipher $E = E_2 \circ E_1 \circ E_0$, input and output difference sets A_0 and A_3
Output: $flag$ ▷ Return the ID result of E over $A_0 \otimes A_3$

Fundamental Stage

$PT[A_0, A_1, H_0, E_0]$ and $PT[A_3, A_2, H_2, E_2^{-1}]$ ▷ Obtain two partition tables
Allocate $I \leftarrow \emptyset$
for $(a_1, a_2) \in A_1 \otimes A_2$ **do**
  if $a_1 \stackrel{E_1}{\nrightarrow} a_2$ **then** ▷ Build a model to determine whether $a_1 \stackrel{E_1}{\rightarrow} a_2$
    $I \leftarrow I \cup \{(a_1, a_2)\}$
  end if
end for
if $I = \emptyset$ **then**
  return $flag = 0$ ▷ E has no ID over $A_0 \otimes A_3$
end if

Enhance Stage

$I_1 = \{a_1 \in A_1 | \, (a_1, a_2) \notin I$ holds for every $a_2 \in A_2\}$
$I_2 = \{a_2 \in A_2 | \, (a_1, a_2) \notin I$ holds for every $a_1 \in A_1\}$
$H_0'[a_1] = H_0[a_1] - \sum_{a \in I_1} H_0[a]$ for any $a_1 \in A_1$
$H_2'[a_2] = H_2[a_2] - \sum_{a \in I_2} H_2[a]$ for any $a_2 \in A_2$
for $(a_1, a_2) \in I$ **do**
  for $(a_0, a_3) \in H_0'[a_1] \otimes H_2'[a_2]$ **do**
    if $a_0 \stackrel{E}{\nrightarrow} a_3$ **then** ▷ Build a model to determine whether $a_0 \stackrel{E}{\rightarrow} a_3$
      return $flag = 1$ ▷ E has at least one ID
    end if
  end for
end for
return $flag = 0$ ▷ E has no ID over $A_0 \otimes A_3$

and $J = \{a_4 \in A_4 | a_4 \stackrel{E_3^{-1}}{\nrightarrow} \exists A_3\}$. In the enhance stage, similarly to partition first implementation strategy in Sect. 3.2.1, we can obtain the reduced hash tables $H_0'[a_1]$ and $H_4'[a_4]$ for any $a_1 \in A_1$ and $a_4 \in A_4$, respectively. Then, for any $a_1 \in I$ and $a_4 \in J$, we further explore whether $H_0'[a_1] \xrightarrow{E_1 \circ E_0} \exists A_2$ and $H_4'[a_4] \xrightarrow{E_3^{-1} \circ E_4^{-1}} \exists A_3$. The whole procedure for obtaining the ID result of E over $A_0 \otimes A_5$ is demonstrated in Algorithm 2.

From **Line 3** and **Line 8** in Algorithm 2, we know that $|A_1| + |A_4|$ differential patterns need to be determined. For example, in **Line 4** of Algorithm 2, we need to determine whether $a_1 \stackrel{E_1}{\rightarrow} \exists A_2$. It should be noted that there is no automatic method for directly modeling this new kind of differential pattern before. For each $a_2 \in A_2$, previous automatic methods [CJF$^+$16,ST17b] will build a model determine whether $a_1 \stackrel{E_1}{\rightarrow} \exists a_2$. Thus, $|A_2|$ models need to be solved. This will greatly increase the complexity of Algorithm 2. In order to tackle this problem, in Sect. 4.2, we propose the definition of *maximal ladder* to guide the selection of a better ladder. Then, the methods for determining a maximal ladder of S-box layer and integrating it into searching model are given. Therefore, we can build only one model to determine whether $a_1 \overset{E_1}{\to} \exists A_2$ effectively.

Algorithm 2 Ladder first implementation strategy

Input: The cipher $E = E_4 \circ \cdots \circ E_0$, input and output difference sets A_0 and A_5
Output: $flag$ ▷ Return the ID result of E over $A_0 \otimes A_5$

Fundamental Stage

$A_2 \overset{E_2}{\to} A_3$, $PT[A_0, A_1, H_0, E_0]$, $PT[A_5, A_4, H_4, E_4^{-1}]$ ▷ ladder and partition tables
Allocate $I \leftarrow \emptyset$ and $J \leftarrow \emptyset$
for $a_1 \in A_1$ **do**
  if $a_1 \overset{E_1}{\nrightarrow} \exists A_2$ **then** ▷ Build a model to determine whether $a_1 \overset{E_1}{\to} \exists A_2$
    $I \leftarrow I \bigcup a_1$
  end if
end for
for $a_4 \in A_4$ **do**
  if $a_4 \overset{E_3^{-1}}{\nrightarrow} \exists A_3$ **then** ▷ Build a model to determine whether $a_4 \overset{E_3^{-1}}{\to} \exists A_3$
    $J \leftarrow J \bigcup a_4$
  end if
end for
if $I = \emptyset$ and $J = \emptyset$ **then**
  return $flag = 0$ ▷ E has no ID over $A_0 \otimes A_5$
end if

Enhance Stage

$H_0'[a_1] = H_0[a_1] - \sum_{a \in A_1 - I} H_0[a]$ for any $a_1 \in A_1$
$H_4'[a_4] = H_4[a_4] - \sum_{a \in A_4 - J} H_4[a]$ for any $a_4 \in A_4$
for $a_1 \in I, a_0 \in H_0'[a_1]$ **do**
  if $a_0 \overset{E_1 \circ E_0}{\nrightarrow} \exists A_2$ **then**
    return $flag = 2$ ▷ Cannot determine whether E has ID
  end if
end for
for $a_4 \in J, a_5 \in H_4'[a_4]$ **do**
  if $a_5 \overset{E_3^{-1} \circ E_4^{-1}}{\nrightarrow} \exists A_3$ **then**
    return $flag = 2$ ▷ Cannot determine whether E has ID
  end if
end for
return $flag = 0$ ▷ E has no ID over $A_0 \otimes A_5$

3.2.3 Dynamic-Ladder-Partition Implementation Strategy Different from the above two strategies, this strategy will determine the ladders and partition tables dynamically. For a cipher $E = E_2 \circ E_1 \circ E_0$, let A_0 and A_3 be the input and output difference sets, respectively. We will dynamically add elements into the ladder $A_1 \otimes A_2$ of E_1 until $A_0 \overset{E_0}{\to} \exists A_1$ and $A_3 \overset{E_2^{-1}}{\to} \exists A_2$ are satisfied or we obtain an ID. Then, we get the ID result of E over $A_0 \otimes A_3$. The whole procedure for obtaining the ID result of the cipher E is demonstrated in Algorithm 3. According to **Line 4** and **Line 13** of Algorithm 3, the elements $a_0 \in A_0$ and $a_3 \in A_3$ are randomly selected. When $flag = 2$, if we want to get a more accurate result, we can call Algorithm 3 again.

Algorithm 3 Dynamic-ladder-partition implementation strategy

Input: The cipher $E = E_2 \circ E_1 \circ E_0$, input and output difference sets A_0 and A_3
Output: $flag$ ▷ Return the ID result of E over $A_0 \otimes A_3$
1: Allocate $A_1 \leftarrow \emptyset$,$A_2 \leftarrow \emptyset$
2: **while** $A_0 \neq \emptyset$ or $A_3 \neq \emptyset$ **do**
3:   **if** $A_0 \neq \emptyset$ **then**
4:     Randomly select an element $a_0 \in A_0$
5:     **if** there exists a_1 satisfying $a_0 \xrightarrow{E_0} a_1$ and $A_1 \cup a_1 \xrightarrow{E_1} A_2$ **then**
6:       $A_0 \leftarrow A_0 - \{a_0 \in A_0 | a_0 \xrightarrow{E_0} a_1\}$ ▷ Remove elements represented by a_1
7:       $A_1 \rightarrow A_1 \bigcup a_1$ ▷ Add element into the set A_1
8:     **else**
9:       **return** $flag = 2$ ▷ Cannot determine whether E has ID
10:     **end if**
11:   **end if**
12:   **if** $A_3 \neq \emptyset$ **then**
13:     Randomly select an element $a_3 \in A_3$
14:     **if** there exists a_2 satisfying $a_3 \xrightarrow{E_2^{-1}} a_2$ and $A_1 \xrightarrow{E_1} A_2 \cup a_2$ **then**
15:       $A_3 \leftarrow A_3 - \{a_3 \in A_3 | a_3 \xrightarrow{E_2^{-1}} a_2\}$ ▷ Remove elements represented by a_2
16:       $A_2 \rightarrow A_2 \bigcup a_2$ ▷ Add element into the set A_2
17:     **else**
18:       **return** $flag = 2$ ▷ Cannot determine whether E has ID
19:     **end if**
20:   **end if**
21:   **if** $A_0 = \emptyset$ and $A_3 = \emptyset$ **then**
22:     **return** $flag = 0$ ▷ E has no ID over $A_0 \otimes A_3$
23:   **end if**
24: **end while**

3.2.4 Comparative Analysis of the Three Strategies We will compare and analyze the above strategies from efficiency and accuracy. Efficiency is about the number of models we need to solve. Accuracy is about whether we can get the ID bound of a cipher. Because the enhance stages of Algorithm 1 and 2 are greatly affected by the properties of specific ciphers and fundamental stages play a more important role in most cases. Thus, only the fundamental stages of Algorithm 1 and 2 participate in the comparison. The comparison data of the three implementation strategies are showed in Table 3.

Table 3. The comparison data of the three implementation strategies

	Algorithm 1	Algorithm 2	Algorithm 3
Cipher	$E = E_2 \circ E_1 \circ E_0$	$E = E_4' \circ \cdots \circ E_1' \circ E_0'$	$E = E_2'' \circ E_1'' \circ E_0''$
Partition	$PT[A_0, A_1, H_0, E_0]$ $PT[A_3, A_2, H_2, E_2^{-1}]$	$PT[A_0', A_1', H_0', E_0']$ $PT[A_5', A_4', H_4', E_4'^{-1}]$	$PT[A_0'', A_1'', H_0'', E_0'']$ $PT[A_3'', A_2'', H_2'', E_2''^{-1}]$
Ladder	$A_1 \overset{E_1}{\to} A_2$	$A_2' \overset{E_2'}{\to} A_3'$	$A_1'' \overset{E_1}{\to} A_2''$
Representative	–	$A_1' \overset{E_1'}{\to} \exists A_2'$ $A_4' \overset{E_3'^{-1}}{\to} \exists A_3'$	–
Models	$\|A_1\| \times \|A_2\|$	$\|A_1'\| + \|A_4'\|$	–

Under normal conditions, all input and output difference sets of the three strategies are partitioned over the same functions which means $E_0 = E_0' = E_0''$ and $E_2 = E_4' = E_2''$. Thus, $|A_1| = |A_1'|$ and $|A_2| = |A_4'|$.

Efficiency Comparison. From Table 3, the number of models need to be solved in Algorithm 1 is $|A_1| \times |A_2|$, while the number of models need to be solved in Algorithm 2 is $|A_1'| + |A_4'|$. Thus, ladder first implementation strategy is more efficient than partition first implementation strategy.

Accuracy Comparison. If we obtain the result $flag = 0$ in the fundamental stage of Algorithm 2, it means that $A_1' \overset{E_1'}{\to} \exists A_2'$ and $A_4' \overset{E_3'^{-1}}{\to} \exists A_3'$. Because $A_2' \otimes A_3'$ is a ladder of E_2', we have $A_1' \xrightarrow{E_3' \circ E_2' \circ E_1'} A_4'$ which means that Algorithm 1 will also return $flag = 0$. Thus, if Algorithm 2 can obtain the ID bound of cipher E, Algorithm 1 must also obtain the ID bound. But the opposition is not necessarily the case. Therefore, partition first implementation strategy is more accurate than ladder first implementation strategy. If the time complexity is affordable, we first choose partition first implementation strategy.

It should be noted that the ladders and partition tables of Algorithm 3 are determined dynamically, it is difficult for us to theoretically evaluate its efficiency and accuracy.

4 The Implementation Technologies for the Framework

4.1 Methods for Determining Representative Set and Partition Table

Because the choices of representative set and partition table will have an important influence on the number of models need to be solved. Previous methods in [HPW22] cannot be applied into large-size S-box (e.g. 32-bit) and cannot guarantee the obtained representative sets and partition tables are optimal representative sets and partition tables defined as following.

Definition 6. (Optimal Representative Set and Partition Table). *For an S-box S, let A be the set of input differences. For a partition table $PT[A, B, H, S]$,* *if the number of elements in the set B is the minimum, we call B the optimal representative set and $PT[A, B, H, S]$ the optimal partition table of A over S.*

To help readers better understand the significance of the above definition, we take Algorithm 1 for example. The number of models need to be solved in fundamental stage of Algorithm 1 is $|A_1| \times |A_2|$. If $PT[A_0, A_1, H_0, E_0]$ and $PT[A_3, A_2, H_2, E_2^{-1}]$ are optimal partition tables, the number of models to be solved in fundamental stage will be minimum. For S-boxes of different sizes, we propose corresponding methods for determining their representative sets and partition tables as following.

4.1.1 The Method for Small-Size S-box

When the size of an S-box is small (e.g. 4-bit or 8-bit), inspired by the method in [ST17a], we propose an automatic method based on MILP to obtain its optimal representative set and partition table. For an S-box S, let A and B be the input and output difference sets, respectively. The overview of our algorithm is as follow. Firstly, for each input difference $a \in A$, we compute the set of output differences that can be the representative of a, denoted as $R[a] = \{b \in B | a \xrightarrow{S} b\}$. Secondly, for each $a \in A$, we construct a constraint such that there must be at least 1 element of $R[a]$ belong to the representative set. Finally, we minimize the number of elements in the representative set under these constraints.

Constraints. For each $b \in B$ we introduce a binary variables v_b, where $v_b = 1$ means that the output difference b is included in the representative set and $v_b = 0$ means that b is not included in the representative set. The only constraint we need is ensuring that each $a \in A$ has at least one representative, which can be represented by the following $|A|$ constraints.

$$\sum_{b \in R[a]} v_b \geq 1, a \in A.$$

Objective Function. Our goal is to find an optimal representative set. Thus, the objective function can be expressed as

$$\text{minimize} \sum_{b \in B} v_b.$$

By solving the above MILP model, we obtain the solutions of $v_b, b \in B$. Thus, the optimal representative set is $B' = \{b \in B | v_b = 1\}$. The whole procedure for obtaining the optimal representative set of S is demonstrated in Algorithm 4.

According to Definition 4 and Definition 6, by removing the overlapping elements among sets $\{a \in A | a \xrightarrow{S} b'\}, b' \in B'$, we can get the optimal partition table $PT[A, B', H, S]$.

4.1.2 The Method for Middle-Size S-box

When we use the method in 4.1.1 to determine the optimal representative set and partition table of middle-

Algorithm 4 The optimal representative set of small-size S-box

Input: The S-box S, input and output difference sets A and B
Output: The optimal representative set B' of A over S

```
Let 𝓜 be an empty MILP model
𝓜.Objective = minimize ∑_{b∈B} v_b                ▷ Set the objective function
for a ∈ A do
    𝓜.addConstr ( ∑_{b∈R[a]} v_b ≥ 1 )           ▷ Set the constraints
end for
𝓜.optimize()                                      ▷ Solve the MILP model
return B' = {b ∈ B | v_b = 1}                     ▷ Obtain the optimal representative set
```

size S-box (e.g. 16-bit), the MILP model are too large to be solved. Thus, we propose a method to solve this problem.

Theorem 6. *For an S-box S, let A and B be the input and output difference sets, respectively. Selecting a subset $A' \subseteq A$, let B' be the optimal representative set of A'. If B' is a representative set of A, then B' is an optimal representative set of A.*

Proof. Let B'' be an optimal representative set of A. Since $A' \subseteq A$, B'' is also the representative set of A'. Because B' is the optimal representative set of A', we have $|B'| \leq |B''|$. When B' is a representative set of A, according to the definition of optimal representative set, B' must be the optimal representative set of A. □

For the small subset $A' \subseteq A$, we can use Algorithm 4 to obtain the optimal representative set B' of A'. If B' is the representative of A, then we obtain an optimal representative set of A. If B' is not the representative of A, we will add the elements which cannot be represented by B' into A'. That is, $A' = A' + \{a \in A | a \overset{S}{\nrightarrow} B'\}$. Using this method, we will keep adding elements into A' until the corresponding B' is the optimal representative set of A. The whole procedure for obtaining an optimal representative set of A over S is demonstrated in Algorithm 5. Using the same method in Sect. 4.1.1, we can get the optimal partition table $PT[A, B', H, S]$ of A over S.

4.1.3 The Method for Large-Size Superbox

When the size of an S-box is large (e.g. 32-bit), it is hard to obtain its optimal representative set. Because most S-boxes of large size are superboxes illustrated in Fig 3, where $s_i, 0 \leq i \leq m-1$ are bijective small-size S-boxes and P is a bijective linear function. In order to construct a representative set with relatively few elements, we propose the following theorem.

Theorem 7. *For an S-box $S = (s_{m-1}||s_{m-2}|| \cdots ||s_0) \circ P \circ (s_{m-1}||s_{m-2}|| \cdots ||s_0)$, let $A = A_{m-1} \otimes A_{m-2} \otimes \cdots \otimes A_0$ and $B = B_{m-1} \otimes B_{m-2} \otimes \cdots \otimes B_0$ be the input and output difference sets, respectively. For each $0 \leq i \leq m-1$, let B'_i be the optimal representative set of A_i over s_i and $B''_i \subseteq B_i$ be the representative of all*

Algorithm 5 The optimal representative set of middle-size S-box

Input: The S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$, input and output difference sets A and B
Output: The optimal representative set B'

Select a subset $A' \subseteq A$ and let $B' = \emptyset$
while B' is not the representative set of A **do**
  Using Algorithm 4 to obtain the optimal representative set B' of A'
  if B' is the representative of A **then**
    return B'
  else
    $A' = A' + \{a \in A | a \overset{S}{\nrightarrow} B'\}$
  end if
end while

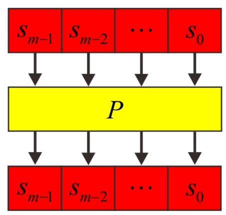


Fig. 3. Large-size superbox

possible differences $\{a | a \in \mathbb{F}_2^n\}$ over s_i, where n is the dimension of s_i. Then, we can use Algorithm 4 to obtain a representative set $C \subseteq B''_{m-1} \otimes B''_{m-2} \otimes \cdots \otimes B''_0$ of $B'_{m-1} \otimes B'_{m-2} \otimes \cdots \otimes B'_0$ over $(s_{m-1} || s_{m-2} || \cdots || s_0) \circ P$. Thus, C is a representative set of A.

Proof. Because $B''_{m-1} \otimes B''_{m-2} \otimes \cdots \otimes B''_0$ is the representative set of $\{a | a \in \mathbb{F}_2^{n \times m}\}$ over $(s_{m-1} || s_{m-2} || \cdots || s_0)$ and $B'_{m-1} \otimes B'_{m-2} \otimes \cdots \otimes B'_0 \xrightarrow{P} \exists \{a | a \in \mathbb{F}_2^{n \times m}\}$, we have $B''_{m-1} \otimes B''_{m-2} \otimes \cdots \otimes B''_0$ is a representative set of $B'_{m-1} \otimes B'_{m-2} \otimes \cdots \otimes B'_0$ over $(s_{m-1} || s_{m-2} || \cdots || s_0) \circ P$. Thus, we must be able to select a representative set $C \subseteq B''_{m-1} \otimes B''_{m-2} \otimes \cdots \otimes B''_0$ of $B'_{m-1} \otimes B'_{m-2} \otimes \cdots \otimes B'_0$ over $(s_{m-1} || s_{m-2} || \cdots || s_0) \circ P$. Because $B'_{m-1} \otimes B'_{m-2} \otimes \cdots \otimes B'_0$ is the representative set of $A_{m-1} \otimes A_{m-2} \otimes \cdots \otimes A_0$ over $(s_{m-1} || s_{m-2} || \cdots || s_0)$, C is a representative set of A over S. $\square$

The representative set C obtained by Theorem 7 may contain redundant representative elements, we need to reduce C further. The whole procedure of obtaining a representative set of large-size superbox S is demonstrated in Algorithm 6. Moreover, using the same method in Sect. 4.1.1, we can get the corresponding partition table $PT[A, C', H, S]$.

Algorithm 6 The representative set of superbox

Input: The S-box $S = (s_{m-1}||s_{m-2}||\cdots||s_0) \circ P \circ (s_{m-1}||s_{m-2}||\cdots||s_0)$, input and output difference sets $A = A_{m-1} \otimes A_{m-2} \otimes \cdots \otimes A_0$ and $B = B_{m-1} \otimes B_{m-2} \otimes \cdots \otimes B_0$

Output: The representative set of A over S

for $0 \le i \le m-1$ **do** ▷ Using Algorithm 4
  Obtain the optimal representative set B'_i of A_i over s_i
  Obtain the optimal representative set B''_i of $\{a | a \in \mathbb{F}_2^n\}$ over s_i
end for
Using Algorithm 4 to obtain the representative set $C \subseteq B''_{m-1} \otimes B''_{m-2} \otimes \cdots \otimes B''_0$ of $B'_{m-1} \otimes B'_{m-2} \otimes \cdots \otimes B'_0$ over $(s_{m-1}||s_{m-2}||\cdots||s_0) \circ P$
Allocate $C' = \emptyset$
while $A \neq \emptyset$ **do**
  Select an element $a \in A$ and $c \in C$ satisfying $a \xrightarrow{S} c$
  $A \leftarrow A - \{a \in A | a \xrightarrow{S} c\}$ ▷ Remove the elements which have been represented
  $C' \leftarrow C' + \{c\}$ and $C \leftarrow C - \{c\}$
end while
return C'

4.2 Methods for Determining Ladder and Integrating it into Model

4.2.1 Method for Determining Ladder When we use Algorithm 2 to evaluate the ID bound, we have to construct a ladder. To guide the selection of ladders, we propose the following theorem.

Theorem 8. *For cipher $E = E_4 \circ E_3 \circ E_2 \circ E_1 \circ E_0$, let $A_2 \otimes A_3$ and $A'_2 \otimes A'_3$ be two ladders of E_2 satisfying $A_2 \otimes A_3 \subseteq A'_2 \otimes A'_3$. When applying Algorithm 2 to E, if we obtain the ID result $flag = 0$ when using ladder $A_2 \otimes A_3$, we can definitely get the ID result $flag = 0$ when using ladder $A'_2 \otimes A'_3$.*

Proof. According to Algorithm 2, only when $a_0 \xrightarrow{E_1 \circ E_0} \exists A_2$ and $a_5 \xrightarrow{E_3^{-1} \circ E_4^{-1}} \exists A_3$ hold for all $a_0 \in A_0, a_5 \in A_5$, the ID result $flag = 0$ can be obtained. Because $A_2 \otimes A_3 \subseteq A'_2 \otimes A'_3$, the conditions $a_0 \xrightarrow{E_1 \circ E_0} \exists A'_2$ and $a_5 \xrightarrow{E_3^{-1} \circ E_4^{-1}} \exists A'_3$ are met. Thus, we can get the ID result $flag = 0$ when using ladder $A'_2 \otimes A'_3$. □

The goal of the paper is to obtain the ID bounds of block ciphers. Compared with ladder $A_2 \otimes A_3$, there is no doubt that $A'_2 \otimes A'_3$ is a better choice. Thus, we propose the following definition.

Definition 7. (Maximal Ladder). Let $A \otimes B$ be a ladder of function f. If there is no other ladder $A' \otimes B'$ of f satisfying $A \otimes B \subseteq A' \otimes B'$, we call $A \otimes B$ a *maximal ladder* of f.

According to Theorem 8, if a ladder $A \otimes B$ is not a maximal ladder, there always exists a better ladder. Thus, when applying Algorithm 2 to ciphers, only maximal ladders are used. Generally, we often use the maximal ladder of an S-box layer.

Theorem 9. (Maximal Ladder of S-box). *Let S be a bijective S-box. For any input difference $a \in \mathbb{F}_2^n$, we can obtain its output difference set, denoted as $DDT_S[a] = \{b \in \mathbb{F}_2^n | a \xrightarrow{S} b\}$. Thus, $A \otimes B$ is a maximal ladder of S if and only if the following conditions are satisfied.*

$$\begin{cases} B = \bigcap_{a \in A} DDT_S[a], \\ A = \bigcap_{b \in B} DDT_{S^{-1}}[b], \end{cases}$$

where S^{-1} is the inverse function of S.

Proof. **Sufficiency.** Because $B = \bigcap_{a \in A} DDT_S[a]$, we have $A \xrightarrow{S} B$ and there is no element $b' \notin B$ satisfying $A \xrightarrow{S} B \bigcup b'$. Similarly, there is no element $a' \notin A$ satisfying $B \xrightarrow{S^{-1}} A \bigcup a'$. According to Theorem 4, $B \xrightarrow{S^{-1}} A \bigcup a'$ is equivalent to $A \bigcup a' \xrightarrow{S} B$. Thus, there does not exist any $b' \notin B$ or $a' \notin A$ satisfying $A \bigcup a' \xrightarrow{S} B$ or $A \xrightarrow{S} B \bigcup b'$. Therefore, $A \otimes B$ is a maximal ladder of S.

Necessity. Because $A \otimes B$ is a ladder of S, we have $B \subseteq \bigcap_{a \in A} DDT_S[a]$. Since $A \xrightarrow{S} \bigcap_{a \in A} DDT_S[a]$ is also a ladder, the maximal ladder $A \otimes B$ must satisfy $B = \bigcap_{a \in A} DDT_S[a]$. According to Theorem 4, $B \otimes A$ is a maximal ladder of S^{-1}. Similarly, we have $A = \bigcap_{b \in B} DDT_{S^{-1}}[b]$. □

Based on the above theorem, we propose a heuristic method to obtain a maximal ladder of S. The whole procedure is demonstrated in Algorithm 7.

Algorithm 7 Heuristic method for determining a maximal ladder of S-box

Input: The bijective S-box S, initial input difference set $A \neq \emptyset$
Output: A maximal ladder of S

```
1: Allocate B ← ∅
2: while 1 do
3:     C = ⋂_{a∈A} DDT_S[a] − B        ▷ The set of elements which can be added into B
4:     Select a subset C' ⊆ C
5:     B ← B + C'                       ▷ Expand the size of B
6:     D = ⋂_{b∈B} DDT_{S^{-1}}[b] − A  ▷ The set of elements which can be added into A
7:     Select a subset D' ⊆ D
8:     A ← A + D'                       ▷ Expand the size of A
9:     if B = ⋂_{a∈A} DDT_S[a] and A = ⋂_{b∈B} DDT_{S^{-1}}[b] then
10:        return A ⊗ B                 ▷ If A ⊗ B is already a maximal ladder of S
11:    end if
12: end while
```

Then, we can use the maximal ladders of small-size S-boxes to construct a maximal ladder of an S-box layer. The method is shown in Theorem 10.

Theorem 10. (Maximal Ladder of an S-box Layer). *Let S be a function comprising of m parallel S-boxes, denoted as $S = s_{m-1} || s_{m-2} || \cdots || s_0$. For each*

$0 \le i \le m-1$, *if* $A_i \otimes B_i$ *is a maximal ladder of* s_i, *then* $\left(\bigotimes_{i=0}^{m-1} A_i\right) \otimes \left(\bigotimes_{i=0}^{m-1} B_i\right)$ *is a maximal ladder of* S.

Proof. Because $A_i \otimes B_i$ is a ladder of s_i, for any $a_i \in A_i$ and $b_i \in B_i$, we have $a_i \xrightarrow{s_i} b_i$. Thus, for any $(a_{m-1}, a_{m-2}, \cdots, a_0) \in \bigotimes_{i=0}^{m-1} A_i$ and $(b_{m-1}, b_{m-2}, \cdots, b_0) \in \bigotimes_{i=0}^{m-1} B_i$, we have $(a_{m-1}, a_{m-2}, \cdots, a_0) \xrightarrow{S} (b_{m-1}, b_{m-2}, \cdots, b_0)$. Therefore, $\left(\bigotimes_{i=0}^{m-1} A_i\right) \otimes \left(\bigotimes_{i=0}^{m-1} B_i\right)$ is a ladder of S.

If $\left(\bigotimes_{i=0}^{m-1} A_i\right) \otimes \left(\bigotimes_{i=0}^{m-1} B_i\right)$ is not a maximal ladder of S, there exists an element $\left(a'_{m-1}, a'_{m-2}, \ldots, a'_0\right) \notin \bigotimes_{i=0}^{m-1} A_i$ or $\left(b'_{m-1}, b'_{m-2}, \ldots, b'_0\right) \notin \bigotimes_{i=0}^{m-1} B_i$ satisfying $\left(\left(a'_{m-1}, a'_{m-2}, \ldots, a'_0\right) \bigcup \bigotimes_{i=0}^{m-1} A_i\right) \otimes \left(\bigotimes_{i=0}^{m-1} B_i\right)$ or $\left(\bigotimes_{i=0}^{m-1} A_i\right) \otimes \left(\left(b'_{m-1}, b'_{m-2}, \ldots, b'_0\right) \bigcup \bigotimes_{i=0}^{m-1} B_i\right)$ is also a ladder of S. Take one of the ladders $\left(\left(a'_{m-1}, a'_{m-2}, \ldots, a'_0\right) \bigcup \bigotimes_{i=0}^{m-1} A_i\right) \otimes \left(\bigotimes_{i=0}^{m-1} B_i\right)$ as an example, for each $0 \le i \le m-1$, we have $a'_i \xrightarrow{s_i} B_i$. Because any $A_i \times B_i, 0 \le i \le m-1$ is a maximal ladder of s_i, we obtain that $a'_i \in A_i$. It is contradictory to $\left(a'_{m-1}, a'_{m-2}, \ldots, a'_0\right) \notin \bigotimes_{i=0}^{m-1} A_i$. Similarly, we can also obtain the contradictory of $\left(b'_{m-1}, b'_{m-2}, \ldots, b'_0\right) \notin \bigotimes_{i=0}^{m-1} B_i$. Therefore, $\left(\bigotimes_{i=0}^{m-1} A_i\right) \otimes \left(\bigotimes_{i=0}^{m-1} B_i\right)$ is a maximal ladder of S. □

4.2.2 Methods for Integrating a Ladder into Searching Model After obtaining a ladder, we should integrate it into searching model (MILP or SAT). For example, in **Line 4** and **Line 9** of Algorithm 2, we need to determine whether $a_1 \xrightarrow{E_1} \exists A_2$ and $a_4 \xrightarrow{E_3^{-1}} \exists A_3$ or not, where $A_2 \otimes A_3$ is a ladder of E_2. It should be noted that there is no automatic method for directly modeling this new kind of differential pattern before. Here, we put forward a solution. Similar to current automatic searching models based on MILP or SAT, we introduce a sequence of variables and constraints satisfying the differential propagation rules. Take $a_1 \xrightarrow{E_1} \exists A_2$ as an example, we can construct a model $\mathcal{M}$ whose solutions are all possible differential characteristics of E_1. Let x and $y = y_{m-1}||y_{m-2}||\cdots||y_0$ be the variables representing the input and output differences of E_1.

When E_2 is a function comprising of m parallel bijective S-boxes, denoted as $E_2 = s_{m-1}||s_{m-2}||\cdots||s_0$. For any $0 \le i \le m-1$, we can construct the maximal ladder of s_i, denoted as $A_{2,i} \times A_{3,i}$. In order to model $a_1 \xrightarrow{E_1} \exists A_2 = A_{2,m-1} \otimes A_{2,m-2} \otimes \cdots \otimes A_{2,0}$, we add the following constraints into $\mathcal{M}$:

$$\mathcal{C} = \begin{cases} x = a_1, \\ y_i \neq d, \text{ where } d \in \{d \in \mathbb{F}_2^{n_i} | d \notin A_{2,i}\}, 0 \le i \le m-1, \end{cases}$$

where n_i is the dimension of s_i.

Then, if the whole model $\mathcal{M} + \mathcal{C}$ is feasible, we have $a_1 \xrightarrow{E_1} \exists A_2$. Otherwise, $a_1 \overset{E_1}{\nrightarrow} \exists A_2$

4.2.3 Exploring Rotation-Equivalence ID Set In [EME22], Erlacher *et al.* exploited the rotational symmetry of ASCON and reduced the number of differential patterns need to be considered. Inspired by their work, we propose the rotation-equivalence ID set defined as following.

Definition 8. (Rotation-Equivalence ID Set). *For a cipher E, let $A^m \subseteq \{a|a \in \mathbb{F}_2^{m\times n}\}$ and $B^m \subseteq \{b|b \in \mathbb{F}_2^{m\times n}\}$ be the input and output difference sets, respectively, where n is the dimension of the elements in A and B. $A^m \otimes B^m$ is called the rotation-equivalence ID set, if it satisfies the following conditions. For any $a \in A^m$, if there exists an output difference $b \in B^m$ satisfying $a \overset{E}{\nrightarrow} b$, then for each $1 \leq l \leq m-1$, there exists an output difference $b_l \in B^m$ satisfying $(a \lll l \times n) \overset{E}{\nrightarrow} b_l$.*

For the rotation-equivalence ID set $A^m \otimes B^m$ of E, we can divide the input difference set A^m into many disjoint subsets as following

$$A^m = \sum_{r\in R} \Omega_r, \tag{6}$$

where $R \subseteq A^m$ and $\Omega_r = \{r \lll l \times n | 0 \leq l \leq m-1\}$. According to Definition 8, all elements in Ω_r have the same result of determining whether E has ID. Thus, for each Ω_r, we only need to consider one element. This will reduce the number of differentials need to be considered. In combinatorics terminology, the subset Ω_r in Eq. (6) is called $|A|$-ary **necklaces** of length m. According to Refield-Pólya theorem [Red27,Pól37], the number of k-ary necklaces of length m is

$$N_k(m) = \frac{1}{m} \sum_{d|m} \varphi(d) \cdot k^{\frac{m}{d}}, \tag{7}$$

where φ is the Euler totient function and d is the divisor of m. For example, the number of 3-ary necklaces of length 4 is

$$N_3(4) = \frac{1}{4}\left(\varphi(1)\cdot 3^{\frac{4}{1}} + \varphi(2)\cdot 3^{\frac{4}{2}} + \varphi(4)\cdot 3^{\frac{4}{4}}\right) = \frac{1}{4}\left(3^4 + 3^2 + 2\times 3\right) = 24.$$

For $A^m \otimes B^m$ of E, there are $|A|^m \times |B|^m$ differential. If $A^m \otimes B^m$ is rotation-equivalence ID set of E, the number of disjoint subsets Ω_r in Eq. (6) is $|R| = N_{|A|}(m)$. Thus, when we evaluate the ID bound of E, only $N_{|A|}(m) \times |B|^m$ differentials need to be considered. Moreover, there is algorithm which can generating necklaces in constant amortized time, see [CRS+00].

5 Applications to SPN Ciphers with Bit-Permutation Linear Layer

In order to improve the hardware efficiency, lightweight block ciphers often use bit-permutation linear layer. The representative algorithms are PRESENT [BKL+07] and GIFT [BPP+17].

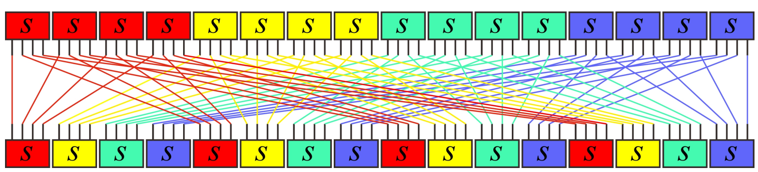
5.1 Application to PRESENT

PRESENT [BKL+07] is an important lightweight cipher. It adopts SPN structure with 64-bit block size through 31 rounds. Each round has three operations: AddRoundKey (XORed with a 64-bit round key), SubBox (16 parallel applications of the same 4-bit S-box, denoted by $S = s^{16||}$), BitPermutation (a bit-wise permutation of 64 bits, denoted as P). PRESENT is a Markov cipher. Under the assumption that the round keys are uniformly random, the AddRoundKey operation can be omitted. Therefore, the round function of PRESENT can be denoted as $R = P \circ S$. An illustration for $S \circ P \circ S$ is shown in Fig. 4(a). By introducing a bit oriented permutation $P_1 = [0, 4, 8, 12, 1, 5, 9, 13, 2, 6, 10, 14, 3, 7, 11, 15]$ and a nibble oriented permutation $P_2 = [0, 4, 8, 12, 1, 5, 9, 13, 2, 6, 10, 14, 3, 7, 11, 15]$, we can get an equivalent representation of $S \circ P \circ S$ as shown in Fig. 4(b). Then,

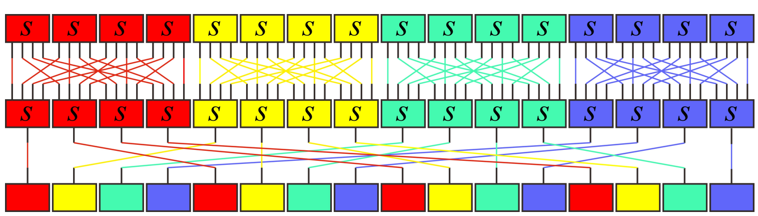
$$S \circ P \circ S = P_2 \circ S \circ (P_1||P_1||P_1||P_1) \circ S.$$

For $(r + 4)$-round PRESENT R^{r+4}, because $P \circ P_2$ is a linear permutation, we omit $P \circ P_2$ in the last round. This will not affect the result of ID bound. Thus,

$$R^{r+4} = \underbrace{S \circ (P_1||P_1||P_1||P_1) \circ S}_{E_2} \circ \underbrace{R^r \circ P \circ P_2}_{E_1} \circ \underbrace{S \circ (P_1||P_1||P_1||P_1) \circ S}_{E_0}.$$



(a) $S \circ P \circ S$ of PRESENT



(b) $P_2 \circ S \circ (P_1||P_1||P_1||P_1) \circ S$ of PRESENT

Fig. 4. The functions of PRESENT

Next, we use Algorithm 5 to determine the optimal representative sets of $s^{4||} \circ P_1 \circ s^{4||}$ and $s^{-4||} \circ P_1^{-1} \circ s^{-4||}$, where $s^{-4||} = s^{-1}||s^{-1}||s^{-1}||s^{-1}$. From Table 4, we know that the number of elements in the optimal representative sets of $s^{4||} \circ P_1 \circ s^{4||}$ and $s^{-4||} \circ P_1^{-1} \circ s^{-4||}$ are 8 and 9, respectively. When applying Algorithm 1 to PRESENT, the number of models needs to be built in fundamental stage is $\left(8^4-1\right)\times\left(9^4-1\right) = 26863200 \approx 2^{24.68}$. After the fundamental stage of Algorithm 1, for 7-round and 8-round PRESENT, there are too many differentials which need to be further determined in enhance stage. Due to the limited storage and computing capacity, we cannot determine whether there exist IDs for 7-round and 8-round PRESENT. Then, we prove that 9-round PRESENT does not exist any ID under the sole condition that round keys are uniformly random.

Table 4. The optimal representative sets for PRESENT

S-box	The optimal representative sets (hexadecimal)
$s^{4\|\|} \circ P_1 \circ s^{4\|\|}$	{`0`, `766`, `d33`, `5060`, `7000`, `9779`, `ccee`, `0300`}
$s^{-4\|\|} \circ P_1^{-1} \circ s^{-4\|\|}$	{`0`, `700`, `97a`, `bb0`, `9000`, `ae55`, `b0d0`, `dddd`, `e7a7`}

5.2 Applications to GIFT

As an improved version of PRESENT, GIFT [BPP+17] is composed of two version: GIFT-64 with 64-bit block size and GIFT-128 with 128-bit block size. The only difference between the two versions is the bit permutation to accommodate twice more bits for GIFT-128. Both two versions are Markov ciphers. Similar to PRESENT, we omit the linear function $P \circ P_2$ in the last round. The $(r+4)$-round GIFT-64 can be written as

$$R^{r+4} = \underbrace{S \circ (P_1||P_1||P_1||P_1) \circ S}_{E_2} \circ \underbrace{R^r \circ P \circ P_2}_{E_1} \circ \underbrace{S \circ (P_1||P_1||P_1||P_1) \circ S}_{E_0}. \quad (8)$$

where $P_1 = [0,5,10,15,12,1,6,11,8,13,2,7,4,9,14,3]$ is a bit oriented permutation and $P_2 = [0,4,8,12,1,5,9,13,2,6,10,14,3,7,11,15]$ is a nibble oriented permutation. Then, we use Algorithm 5 to determine the optimal representative sets of $s^{4||} \circ P_1 \circ s^{4||}$ and $s^{-4||} \circ P_1^{-1} \circ s^{-4||}$ shown in Table 5. When applying Algorithm 1 to GIFT-64. the number of models needs to be built in fundamental stage is $\left(9^4-1\right)\times\left(8^4-1\right) = 26863200 \approx 2^{24.68}$. After the fundamental stage of Algorithm 1, for 7-round GIFT64, there are too many differentials which need to be further determined in enhance stage. Due to the limited storage and computing capacity, we cannot determine whether there exist IDs for 7-round GIFT64. Then, we prove that 8-round GIFT-64 does not exist any ID under the sole assumption that round keys are uniformly random.

For GIFT-128, if we apply Algorithm 1 to it, the number of models need to be built in the fundamental stage is about $(9^8-1)\times(8^8-1) \approx 2^{49.36}$ which is not affordable. Thus, we will use Algorithm 2 to evaluate its ID bound. For GIFT-128, when we omit the linear function $P \circ P_2$ in the last round, (r_1+r_2+5)-round

Table 5. The optimal representative sets for GIFT-64 and GIFT-128

S-box	The optimal representative set (hexadecimal)
$s^{4\|\|} \circ P_1 \circ s^{4\|\|}$	{`0`, `505`, `55f`, `f35`, `350f`, `50f7`, `5f09`, `9d9d`, `b750`}
$s^{-4\|\|} \circ P_1^{-1} \circ s^{-4\|\|}$	{`0`, `d`, `f9`, `d00`, `7dda`, `9b00`, `cf9c`, `fccd`}

GIFT-128 can be written as

$$R^{r_1+r_2+5} = \underbrace{S \circ P_1^{8||} \circ S}_{E_4} \circ \underbrace{R^{r_2} \circ P}_{E_3} \circ \underbrace{S}_{E_2} \circ \underbrace{R^{r_1} \circ P \circ P_2}_{E_1} \circ \underbrace{S \circ P_1^{8||} \circ S}_{E_0}. \tag{9}$$

where $P_1 = [0, 5, 10, 15, 12, 1, 6, 11, 8, 13, 2, 7, 4, 9, 14, 3]$ is a bit oriented permutation (same with that in GIFT-64) and $P_2 = [0, 8, 16, 24, 1, 9, 17, 25, 2, 10, 18, 26, 3, 11, 19, 27, 4, 12, 20, 28, 5, 13, 21, 29, 6, 14, 22, 30, 7, 15, 23, 31]$ is a nibble oriented permutation. Then, we use Algorithm 7 to find a maximal lader $\{1, 3, 7\} \otimes \{5, 8, 11, 12\}$ of the 4-bit S-box used in GIFT-128. According to Theorem 10, the maximal ladder of S is $\{1, 3, 7\}^{16} \otimes \{5, 8, 11, 12\}^{16}$. When we apply Algorithm 2 to $(r_1 + r_2 + 5)$-round GIFT-128, the number of models need to be built in fundamental stage is $\left(9^8 - 1\right) + \left(8^8 - 1\right) = 59823935 \approx 2^{25.83}$. By setting $r_1 = 4$ and $r_2 = 3$, we prove that 12-round GIFT-128 does not exist any ID under the sole assumption that round keys are uniformly random.

6 Applications to SPN Ciphers with Non-Bit-Permutation Linear Layer

6.1 Applications to Rijndael

Rijndael [DR02] was designed by Daemen and Rijmen in 1998. According to block size, Rijndael can be divided into Rijndael-128, Rijndael-160, Rijndael-192, Rijndael-224 and Rijndael-256. The 128-bit block size version Rijndael-128 was selected as the AES. For Rijndael-$32n$, $n \in \{4, 5, 6, 7, 8\}$, the state is viewed as $4 \times n$ rectangle array of 8-bit words. The round function of Rijndael-$32n$ consists of the following four operations: SubBox ($4 \times n$ parallel applications of the same 8-bit Sbox, denoted as $S = s^{4\times n||}$), ShiftRow (a byte transposition that cyclically shifts the rows of the state over different offsets, denoted as SR), MixColumn (a linear matrix M is multiplied to each column of the state, denoted as MC), AddRoundKey (XORed with a $32n$-bit round key). All versions of Rijndael are Markov ciphers. When the round keys are uniformly random, we do not need to consider the AddRoundKey operation. Therefore, the round function of Rijndael-$32n$ can be denoted as $R = MC \circ SR \circ S$. Because SR and MC are linear operations, we omit SR operation of the first round and the $MC \circ SR$ operation of the last round. This will not affect the result of ID bound. For $(r + 4)$-round Rijndael-$32n$, we have

$$R^{r+4} = \underbrace{S \circ MC \circ S}_{E_2} \circ \underbrace{SR \circ R^r \circ MC \circ SR}_{E_1} \circ \underbrace{S \circ MC \circ S}_{E_0}. \tag{10}$$

The functions E_0 and E_2^{-1} of Rijndael-32n can be seen as n parallel 32-bit superboxes $s^{4||} \circ M \circ s^{4||}$ and $s^{-4||} \circ M^{-1} \circ s^{-4||}$, respectively. Next, we use Algorithm 6 to determine the representative sets of $s^{4||} \circ M \circ s^{4||}$ and $s^{-4||} \circ M^{-1} \circ s^{-4||}$. From Table 6, we know that both the numbers of elements in the representative sets of $s^{4||} \circ MC \circ s^{4||}$ and $s^{-4||} \circ M^{-1} \circ s^{-4||}$ are 2. Then, we explore the rotation-equivalence ID sets of Rijndael-32n shown in Theorem 11.

Theorem 11. *For Rijndael-32n, let a_1 and a_2 be the input and output differences of E_1, respectively. If $a_1 \overset{E_1}{\nrightarrow} a_2$, then $SR_i\left(a_1\right) \overset{E_1}{\nrightarrow} SR_i\left(a_2\right)$ holds for all $i \in \{1, 2, \ldots, n-1\}$, where SR_i means cyclically shifting every row of the state over i bytes.*

Proof. According to the definitions of SR, MC and S, we have the following equations

$$\begin{cases} SR \circ SR_i = SR_i \circ SR \\ MC \circ SR_i = SR_i \circ MC \\ S \circ SR_i = SR_i \circ S \end{cases}$$

Thus, $a_1 \overset{E_1}{\nrightarrow} a_2$ is equivalent to $SR_i\left(a_1\right) \overset{E_1}{\nrightarrow} SR_i\left(a_2\right), i \in \{1, 2, \ldots, n-1\}$. □

Table 6. The representative sets of Rijndael-32n

S-box	The representative set (hexadecimal)
$s^{4\|\|} \circ M \circ s^{4\|\|}$	{0, `f8f9f9f9`}
$s^{-4\|\|} \circ M^{-1} \circ s^{-4\|\|}$	{0, `f8faf8f8`}

We applying Algorithm 1 to Rijndael-32n. According to Sect. 4.2.3, the number of models need to be built in fundamental stage is $\left(N_2\left(n\right)-1\right) \times \left(2^n-1\right)$. Then, we prove that 6-round AES (Rijndael-128), 6-round Rijndael-160, 7-round Rijndael-192, 7-round Rijndael-224, 7-round Rijndael-256 do not have any ID under the sole assumption that round keys are uniformly random.

Because the longest known ID of AES (Rijndael-128) is 4 round, the security bound obtained by us has room for improvement. Therefore, we apply Algorithm 3 to AES. The specific process is as follow. Similarly to the above analysis, 5-round AES can be written as,

$$R^5 = \underbrace{S \circ MC \circ S}_{E_2} \circ \underbrace{SR \circ MC \circ SR \circ S \circ MC \circ SR}_{E_1} \circ \underbrace{S \circ MC \circ S}_{E_0}. \tag{11}$$

Let $A_0 = A_{0,3} \otimes A_{0,2} \otimes A_{0,1} \otimes A_{0,0}$ and $A_3 = A_{3,3} \otimes A_{3,2} \otimes A_{3,1} \otimes A_{3,0}$ be the sets of all nonzero input and output differences of AES, respectively. Thus, the whole search space $A_0 \otimes A_3$ can be divided into the following $15 \times 15 = 225$ disjoint subsets.

$$A_0 \otimes A_3 = \sum_{(i_0,i_1,i_2,i_3) \in \mathbb{F}_2^{4*}, (j_0,j_1,j_2,j_3) \in \mathbb{F}_2^{4*}} \left[A_{0,3}\right]^{i_3} \otimes \cdots \otimes \left[A_{0,0}\right]^{i_0} \otimes \left[A_{3,3}\right]^{j_3} \otimes \cdots \otimes \left[A_{3,0}\right]^{j_0}$$

where $\mathbb{F}_2^{4*} = \{a \in \mathbb{F}_2^4 | a \neq 0\}$ is the set of all nonzero 4-bit vectors. For any $i \in \{0,3\}$ and $m \in \{0,1,2,3\}$, $[A_{i,m}]^0 = \{0 \in \mathbb{F}_2^{32}\}$ be the set of only 32-bit zero difference and $[A_{i,m}]^1 = \{a \in \mathbb{F}_2^{32} | a \neq 0\}$ is the set of all nonzero 32-bit differences. Moreover, according to Theorem 11, we only need to consider $(N_2(4) - 1) \times (2^4 - 1) = 75$ disjoint subsets.

For any of the above subsets, we select $a_0 = (a_{0,3}, a_{0,2}, a_{0,1}, a_{0,0}) \in [A_{0,3}]^{i_3} \otimes \cdots \otimes [A_{0,0}]^{i_0}$ and $a_3 = (a_{3,3}, a_{3,2}, a_{3,1}, a_{3,0}) \in [A_{3,3}]^{j_3} \otimes \cdots \otimes [A_{3,0}]^{j_0}$ and build a model to obtain $a_1 = (a_{1,3}, a_{1,2}, a_{1,1}, a_{1,0})$ and $a_2 = (a_{2,3}, a_{2,2}, a_{2,1}, a_{2,0})$ satisfying $a_0 \xrightarrow{E_0} a_1$, $a_1 \xrightarrow{E_1} a_2$ and $a_3 \xrightarrow{E_2^{-1}} a_2$. If $[A_{0,3}]^{i_3} \otimes \cdots \otimes [A_{0,0}]^{i_0} \xrightarrow{E_0} a_1$ and $[A_{3,3}]^{j_3} \otimes \cdots \otimes [A_{3,0}]^{j_0} \xrightarrow{E_2^{-1}} a_2$, all the differentials in subset $[A_{0,3}]^{i_3} \otimes \cdots \otimes [A_{0,0}]^{i_0} \otimes [A_{3,3}]^{j_3} \otimes \cdots \otimes [A_{3,0}]^{j_0}$ over E are possible.

The method for verifying $[A_{0,3}]^{i_3} \otimes \cdots \otimes [A_{0,0}]^{i_0} \xrightarrow{E_0} a_1$ and $[A_{3,3}]^{j_3} \otimes \cdots \otimes [A_{3,0}]^{j_0} \xrightarrow{E_2^{-1}} a_2$ is as following. Take $[A_{0,3}]^{i_3} \otimes \cdots \otimes [A_{0,0}]^{i_0} \xrightarrow{E_0} a_1$ as an example, we just need to verify whether $[A_{0,m}]^{i_m} \xrightarrow{s^{4||} \circ M \circ s^{4||}} a_{1,m}$ holds for all $m = 0,1,2,3$. For any i_m, if $i_m = 0$, we only need to verify 1 difference and if $i_m = 1$, we have to verify $2^{32} - 1$ input differences in $[A_{0,m}]^{i_m}$. In order to improve the success rate, if $i_m = 1$, we add a constrain to $a_{1,m}$ that every byte of $a_{1,m}$ is nonzero. After verifying all the disjoint subsets, we prove that 5-round AES do not have any ID under the sole assumption that round keys are uniformly random.

6.2 Application to Midori64

Midori64 is a lightweight SPN block cipher with 64-bit block size proposed at ASIACRYPT 2015 [BBI+15]. Each round function consists of the following four operations: SubBox (16 parallel applications of the same 4-bit Sbox, denoted as $S = s^{16||}$), PermuteNibbles (permutation is applied on the nibble positions of the state, denoted as PN), MixColumn (an involutory binary matrix M is multiplied to each column of the state, denoted as MC), AddRoundKey (XORed with a 64-bit round key). Midori64 is a Markov cipher. When the round keys are uniformly random, we do not need to consider the AddRoundKey operation. Therefore, the round function of Midori64 can be denoted as $R = MC \circ PN \circ S$. Because PN and MC are linear operations, we omit PN operation of the first round and the $MC \circ PN$ operation of the last round. This will not affect the result of ID bound. For $(r+4)$-round Midori64, we have

$$R^{r+4} = \underbrace{S \circ MC \circ S}_{E_2} \circ \underbrace{PN \circ R^r \circ MC \circ PN}_{E_1} \circ \underbrace{S \circ MC \circ S}_{E_0}. \tag{12}$$

The functions E_0 and E_2^{-1} of Midori64 can be seen as 4 parallel 16-bit S-boxes $s^{4||} \circ M \circ s^{4||}$ and $s^{-4||} \circ M^{-1} \circ s^{-4||}$, respectively. Next, we use Algorithm 6 to determine the optimal representative sets of $s^{4||} \circ M \circ s^{4||}$ and $s^{-4||} \circ M^{-1} \circ s^{-4||}$ shown in Table 7. When we apply Algorithm 1 to $(r+4)$-round Midori64, the number of fundamental models we need to solve is $(8^4-1) \times (8^4-1) = 16769025 \approx 2^{24}$. Then, we prove that 10-round Midori64 does not have any ID under the sole assumption that round keys are uniformly random.

Table 7. The optimal representative sets of Midori64

S-box	The optimal representative set (hexadecimal)
$s^{4\|\|} \circ M \circ s^{4\|\|}$	{0, 66e, 4e9b, 660e, 6e66, b03b, e660, eb19}
$s^{-4\|\|} \circ M^{-1} \circ s^{-4\|\|}$	{0, 999, 4404, e0ee, e660, ec1e, ecb1, ee6e}

7 Conclusion

In this paper, a series of methods for bounding the length of IDs of SPN block ciphers are proposed. Our methods are widely applicable. We prove that 9-round PRESENT, 8-round GIFT-64, 12-round GIFT-128, 5-round AES, 6-round Rijndael-160, 7-round Rijndael-192, 7-round Rijndael-224, 7-round Rijndael-256 and 10-round Midori64 do not have any ID under the sole assumption that round keys are uniformly random. This is of great significance for evaluating the security of SPN block ciphers against ID attack. However, for some ciphers, there still exist a gap between the ID bounds and the longest known IDs. For example, the longest known ID of PRESENT is 6 rounds, while the ID bound obtained by our method is 9 rounds. How to reduce the gap between the longest known ID and ID bound is our future work.

References

BBI+15. Subhadeep Banik, Andrey Bogdanov, Takanori Isobe, Kyoji Shibutani, Harunaga Hiwatari, Toru Akishita, and Francesco Regazzoni. Midori: A block cipher for low energy. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*, volume 9453 of *Lecture Notes in Computer Science*, pages 411–436. Springer, 2015.

BBS99. Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*, pages 12–23. Springer, 1999.

BC20. Christina Boura and Daniel Coggia. Efficient MILP modelings for sboxes and linear layers of SPN ciphers. *IACR Trans. Symmetric Cryptol.*, 2020(3):327–361, 2020.

BJK+16. Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. The SKINNY family of block ciphers and its low-latency variant MANTIS. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 123–153. Springer, 2016.

BKL+07. Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: an ultra-lightweight block cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, 2007.

BPP+17. Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. GIFT: A small present - towards reaching the limit of lightweight encryption. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, volume 10529 of *Lecture Notes in Computer Science*, pages 321–345. Springer, 2017.

CJF+16. Tingting Cui, Keting Jia, Kai Fu, Shiyao Chen, and Meiqin Wang. New automatic search tool for impossible differentials and zero-correlation linear approximations. *IACR Cryptol. ePrint Arch.*, page 689, 2016.

CJZ+17. Ting Cui, Chenhui Jin, Bin Zhang, Zhuo Chen, and Guoshuang Zhang. Searching all truncated impossible differentials in SPN. *IET Inf. Secur.*, 11(2):89–96, 2017.

CR15. Anne Canteaut and Joëlle Roué. On the behaviors of affine equivalent sboxes regarding differential and linear attacks. In Elisabeth Oswald and Marc Fischlin, editors, *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*, volume 9056 of *Lecture Notes in Computer Science*, pages 45–74. Springer, 2015.

CRS+00. Kevin Cattell, Frank Ruskey, Joe Sawada, Micaela Serra, and C. Robert Miers. Fast algorithms to generate necklaces, unlabeled necklaces, and irreducible polynomials over GF(2). *J. Algorithms*, 37(2):267–282, 2000.

DR02. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.

EME22. Johannes Erlacher, Florian Mendel, and Maria Eichlseder. Bounds for the security of ascon against differential and linear cryptanalysis. *IACR Trans. Symmetric Cryptol.*, 2022(1):64–87, 2022.

HLJ+20. Xichao Hu, Yongqiang Li, Lin Jiao, Shizhu Tian, and Mingsheng Wang. Mind the propagation of states - new automatic search tool for impossible differentials and impossible polytopic transitions. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I*, volume 12491 of *Lecture Notes in Computer Science*, pages 415–445. Springer, 2020.

HPW22. Kai Hu, Thomas Peyrin, and Meiqin Wang. Finding all impossible differentials when considering the DDT. *IACR Cryptol. ePrint Arch.*, page 1034, 2022.

JP07. Jorge Nakahara Jr. and Ivan Carlos Pavão. Impossible-differential attacks on large-block rijndael. In Juan A. Garay, Arjen K. Lenstra, Masahiro Mambo, and René Peralta, editors, *Information Security, 10th Interna-*

tional Conference, ISC 2007, Valparaíso, Chile, October 9-12, 2007, Proceedings, volume 4779 of *Lecture Notes in Computer Science*, pages 104–117. Springer, 2007.

KHS+03. Jongsung Kim, Seokhie Hong, Jaechul Sung, Changhoon Lee, and Sangjin Lee. Impossible differential cryptanalysis for block cipher structures. In Thomas Johansson and Subhamoy Maitra, editors, *Progress in Cryptology - INDOCRYPT 2003, 4th International Conference on Cryptology in India, New Delhi, India, December 8-10, 2003, Proceedings*, volume 2904 of *Lecture Notes in Computer Science*, pages 82–96. Springer, 2003.

Knu98. Lars R. Knudsen. Deal - a 128-bit block cipher. *Technical report, Department of Informatics, University of Bergen, Norway*, 1998.

LLWG14. Yiyuan Luo, Xuejia Lai, Zhongming Wu, and Guang Gong. A unified method for finding impossible differentials of block cipher structures. *Inf. Sci.*, 263:211–220, 2014.

LMM91. Xuejia Lai, James L. Massey, and Sean Murphy. Markov ciphers and differential cryptanalysis. In Donald W. Davies, editor, *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, volume 547 of *Lecture Notes in Computer Science*, pages 17–38. Springer, 1991.

MDRM10. Hamid Mala, Mohammad Dakhilalian, Vincent Rijmen, and Mahmoud Modarres-Hashemi. Improved impossible differential cryptanalysis of 7-round AES-128. In Guang Gong and Kishan Chand Gupta, editors, *Progress in Cryptology - INDOCRYPT 2010 - 11th International Conference on Cryptology in India, Hyderabad, India, December 12-15, 2010. Proceedings*, volume 6498 of *Lecture Notes in Computer Science*, pages 282–291. Springer, 2010.

MWGP11. Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and linear cryptanalysis using mixed-integer linear programming. In Chuankun Wu, Moti Yung, and Dongdai Lin, editors, *Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011. Revised Selected Papers*, volume 7537 of *Lecture Notes in Computer Science*, pages 57–76. Springer, 2011.

Pól37. G. Pólya. Kombinatorische Anzahlbestimmungen für Gruppen, Graphen und chemische Verbindungen. *Acta Mathematica*, 68(none):145 – 254, 1937.

Red27. J. Howard Redfield. The theory of group-reduced distributions. *American Journal of Mathematics*, 49(3):433–455, 1927.

SGWW20. Ling Sun, David Gérault, Wei Wang, and Meiqin Wang. On the usage of deterministic (related-key) truncated differentials and multidimensional linear approximations for SPN ciphers. *IACR Trans. Symmetric Cryptol.*, 2020(3):262–287, 2020.

SHW+14. Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma, and Ling Song. Automatic security evaluation and (related-key) differential characteristic search: Application to simon, present, lblock, DES(L) and other bit-oriented block ciphers. In Palash Sarkar and Tetsu Iwata, editors, *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*, volume 8873 of *Lecture Notes in Computer Science*, pages 158–178. Springer, 2014.

SLG+16. Bing Sun, Meicheng Liu, Jian Guo, Vincent Rijmen, and Ruilin Li. Provable security evaluation of structures against impossible differential and zero correlation linear cryptanalysis. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 196–213. Springer, 2016.

ST17a. Yu Sasaki and Yosuke Todo. New algorithm for modeling s-box in MILP based differential and division trail search. In Pooya Farshim and Emil Simion, editors, *Innovative Security Solutions for Information Technology and Communications - 10th International Conference, SecITC 2017, Bucharest, Romania, June 8-9, 2017, Revised Selected Papers*, volume 10543 of *Lecture Notes in Computer Science*, pages 150–165. Springer, 2017.

ST17b. Yu Sasaki and Yosuke Todo. New impossible differential search tool from design and cryptanalysis aspects - revealing structural properties of several ciphers. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, volume 10212 of *Lecture Notes in Computer Science*, pages 185–215, 2017.

WJ19. Qian Wang and Chenhui Jin. More accurate results on the provable security of AES against impossible differential cryptanalysis. *Des. Codes Cryptogr.*, 87(12):3001–3018, 2019.

WJ21. Qian Wang and Chenhui Jin. Bounding the length of impossible differentials for SPN block ciphers. *Des. Codes Cryptogr.*, 89(11):2477–2493, 2021.

WW12. Shengbao Wu and Mingsheng Wang. Automatic search of truncated impossible differentials for word-oriented block ciphers. In Steven D. Galbraith and Mridul Nandi, editors, *Progress in Cryptology - INDOCRYPT 2012, 13th International Conference on Cryptology in India, Kolkata, India, December 9-12, 2012. Proceedings*, volume 7668 of *Lecture Notes in Computer Science*, pages 283–302. Springer, 2012.

ZWP+08. Lei Zhang, Wenling Wu, Je Hong Park, Bonwook Koo, and Yongjin Yeom. Improved impossible differential attacks on large-block rijndael. In Tzong-Chen Wu, Chin-Laung Lei, Vincent Rijmen, and Der-Tsai Lee, editors, *Information Security, 11th International Conference, ISC 2008, Taipei, Taiwan, September 15-18, 2008. Proceedings*, volume 5222 of *Lecture Notes in Computer Science*, pages 298–315. Springer, 2008.