Collision Attacks on SPONGENT with Grouping Method

Keita Toyama[1], Kosei Sakamoto[2,3], and Takanori Isobe[3]

[1] University of Hyogo, Kobe, Japan. `toyama.uoh.hyogo@gmail.com`
[2] Mitsubishi Electric Corporation, Kamakura, Japan. `sakamoto.kosei@dc.mitsubishielectric.co.jp`
[3] Osaka University, Osaka, Japan. `takanori.isobe@ist.osaka-u.ac.jp`

Abstract. `SPONGENT` is a lightweight hash function based on the sponge construction, featuring a `PRESENT`-like round function. Although it is included in the ISO standard for lightweight cryptography, no third-party analysis of collision attacks has been conducted. This lack of analysis is primarily due to the inherent difficulty of identifying value pairs that satisfy differential characteristics in sponge-construction-based hash functions. In this paper, we propose a novel method to efficiently identify value pairs that satisfy differential characteristics for keyless permutations. To address the existing problems, we introduce a grouping method that classifies input variables into categories such as free bits, fixed bits, and interdependent groups. This categorization facilitates efficient computation of degrees of freedom by solving for valid states within each group independently. We apply this method to variants of `SPONGENT` and, for the first time, demonstrate collision and semi-free-start collision attacks on reduced variants.

Keywords: `SPONGENT`· SAT · differential attacks · collision

1 Introduction

1.1 Background

In recent years, lightweight cryptography for resource-constrained environments, such as IoT devices, has gained significant attention. While research on lightweight block ciphers is very active, the security evaluation of lightweight hash functions remains insufficient, particularly for sponge-based schemes such as `Quark` [2], `PHOTON` [12] and `SPONGENT` [7]. Although these schemes are included in the ISO standard for lightweight cryptography [13], it is surprising that no third-party evaluations of collision attacks, which is one of the most critical properties of a hash function, have been conducted. Existing research has only focused on distinguishing attacks targeting the underlying permutations [1, 14, 19, 27, 30, 31].

`SPONGENT` [7] is a sponge-based lightweight hash function proposed by Bogdanov et al. It is designed using a `PRESENT`-like primitive [8] and offers 13 variants with different security levels, allowing users to select an option based on resource constraints and specific security priorities.

Regarding security evaluation, Abdelraheem showed differential and linear distinguishers for the underlying permutations [1].

Zhang and Liu further performed distinguishing attacks and preimage attacks through truncated differential attacks with the meet-in-the-middle technique [31]. However, no effective evaluation of collision resistance has been conducted so far except designer's evaluation [7]. This lack of evaluation arises from the inherent difficulty of identifying value pairs that satisfy differential characteristics in sponge-construction-based hash functions. The difficulty is primarily due to the absence of key-insertion mechanisms in their underlying permutations. This imposes taking the dependency between values and differences in S-boxes over multiple rounds into account, making obtaining enough degrees of freedom (DoF) quite complex. Indeed, Liu et al. demonstrated that many differential characteristics in the previous studies on sponge-based hash functions are not valid [17].

1.2 Contribution

In this paper, to address the problem of collision attacks on sponge-based hash function, we present a new method to efficiently identify value pairs that satisfy a given differential characteristics for key-less permutations. Our method focuses on modeling dependencies between value and difference transitions, drawing from Liu et al.'s approach [17]. A truth table representing these transitions is constructed and then simplified using logic minimization tools like Espresso. Our contribution is summarized as follow.

- We propose an approach to categorize input variables based on their dependencies. These variables are classified into free bits, which are independent and maximize the DoF; fixed bits, which are constrained to specific values; and interdependent groups, which are analyzed incrementally using SAT solvers. This categorization ensures efficient computation of DoF by solving for valid states within each group independently. By integrating the solutions from these groups, we determine the total DoF, which is crucial for assessing the feasibility of collision attacks on sponge-construction-based hash functions.
- We apply our method to several variants of `SPONGENT`. Table 1 shows the summary of our results. We obtain collision and semi-free-start collision attacks on reduced variants of `SPONGENT` for the first time. To verify our approach, we present actual collision pairs for several variants. It should be noted that the results of our attacks do not immediately affect the security of Spongent as there are still enough security margin.

2 Preliminaries

In this section, we explain differential cryptanalysis and automatic exploration using SAT solvers. Finally, we outline the specifications of `SPONGENT`, the crypto-

Table 1: Summary of collision attack results. Rounds marked with * indicate actual collision pairs found. †: invalid attack as it is less efficient than generic attack

	Collision		Semi-free-start		Ref	Full round	Collision security
	Rounds	Time	Rounds	Time			
SPONGENT-88/80/8	-	-	8*	2^{17}	Ours	45	2^{40}
	6†	$2^{55.2}$	-	-	[6]		
SPONGENT-88/176/88	9	2^{42}	9*	2^{33}	Ours	135	2^{44}
	8*	2^{28}					
SPONGENT-128/128/8	-	-	8*	2^{22}	Ours	70	2^{64}
SPONGENT-128/256/128	12	2^{58}	14	2^{63}	Ours	195	2^{64}
	8*	2^{28}					
SPONGENT-160/160/16	-	-	7	2^{47}	Ours	90	2^{80}
SPONGENT-160/160/80	11	2^{61}	14	2^{78}	Ours	120	2^{80}
SPONGENT-160/320/160	11	2^{56}	13	2^{79}	Ours	240	2^{80}
	8*	2^{28}					
SPONGENT-224/224/16	-	-	8	2^{91}	Ours	120	2^{112}
SPONGENT-224/224/112	10	2^{60}	11	2^{62}	Ours	170	2^{112}
SPONGENT-224/448/224	10	2^{47}	11	2^{57}	Ours	340	2^{112}
	8*	2^{21}					
SPONGENT-256/256/16	-	-	11	2^{110}	Ours	140	2^{128}
SPONGENT-256/256/128	15	2^{79}	15	2^{75}	Ours	195	2^{128}
SPONGENT-256/512/256	15	2^{73}	15	2^{73}	Ours	385	2^{128}
	9*	2^{28}					

graphic hash function analyzed in this paper, and discuss limitations of existing evaluations.

2.1 Specification of SPONGENT

SPONGENT is a lightweight hash function with a sponge-based structure, consisting of 13 variants [7]. Each variant is specified by its hash size n, capacity c, and rate r, and is denoted as SPONGENT-$n/c/r$. The size of the internal state, $b = r + c \geq n$, is called the width. The security requirement for collision attacks is $\min(2^{n/2}, 2^{c/2})$. The specifications of each variant are shown in Table 2.

The sponge construction follows an iterative design with three phases. In the initialization phase, the message is padded to a multiple of r bits. During the absorbing phase, each r-bit message block is XORed with the first r bits of the state. Finally, in the squeezing phase, an n-bit output is obtained. The overall view is shown in Fig. 1.

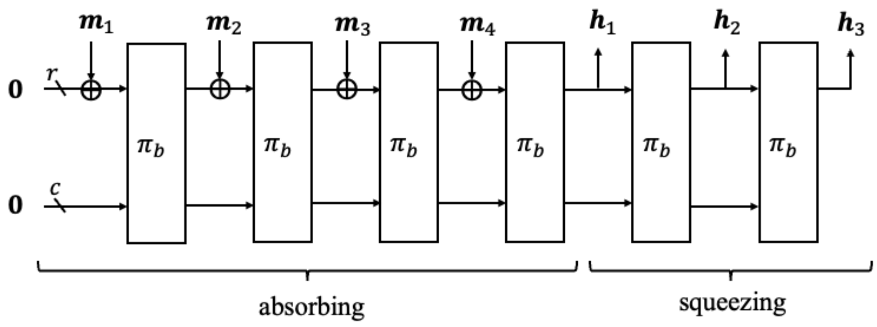

Fig. 1: Sponge construction based on a b-bit permutation π_b with capacity c bits and rate r bits. m_i are r-bit message blocks. h_i are parts of the hash value.

The round function uses an SPN (Substitution-Permutation Network) structure. The round function consists of the following three operations:

– **sBoxLayer**: The S-box operates on 4-bit inputs and produces 4-bit outputs, and it is applied in parallel $b/4$ times. The operation of the S-box in hexadecimal notation is shown as follows.

$$S[\cdot] = \{\texttt{0xE}, \texttt{0xD}, \texttt{0xB}, \texttt{0x0}, \texttt{0x2}, \texttt{0x1}, \texttt{0x4}, \texttt{0xF}, \texttt{0x7}, \texttt{0xA}, \texttt{0x8}, \texttt{0x5}, \texttt{0x9}, \texttt{0xC}, \texttt{0x3}, \texttt{0x6}\}$$

The differential characteristics and internal states are interpreted with the leftmost bit as the least significant bit and represented in hexadecimal notation in this paper.

– **pLayer**: This is an extension of the (inverse) presentbit-permutation. The bit at position j of the state is moved to the bit position *P(j)*. Bit permutation is as shown in Fig. 2.

$$P_b(j) = \begin{cases} j \cdot b/4 \mod (b-1), & \text{if } j \in \{0, \ldots, b-2\} \\ b-1, & \text{if } j = b-1. \end{cases}$$

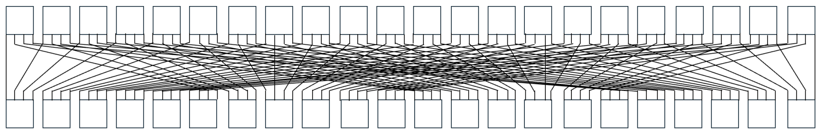

Fig. 2: The bit permutation layer of SPONGENT-88/80/8

– **lCounter**: A round-dependent constant is XORed with the state. The left end of the state is XORed with the round constant, and the right end is XORed with the round constant's bit-reversed value. These round constants are generated by an LFSR (Linear Feedback Shift Register). An LFSR is clocked once each time its state is used, ultimately reaching a final state where all bits are set to 1.

Table 2: Parameters of `SPONGENT`

	n	b	c	r	Rounds	Collision security (bit)
`SPONGENT`-88/80/8	88	88	80	8	45	40
`SPONGENT`-88/176/88	88	264	176	88	135	44
`SPONGENT`-128/128/8	128	136	128	8	70	64
`SPONGENT`-128/256/128	128	384	256	128	195	64
`SPONGENT`-160/160/16	160	176	160	16	90	80
`SPONGENT`-160/160/80	160	240	160	80	120	80
`SPONGENT`-160/320/160	160	480	320	160	240	80
`SPONGENT`-224/224/16	224	240	224	16	120	112
`SPONGENT`-224/224/112	224	336	224	112	170	112
`SPONGENT`-224/448/224	224	672	448	224	340	112
`SPONGENT`-256/256/16	256	272	256	16	140	128
`SPONGENT`-256/256/128	256	384	256	128	195	128
`SPONGENT`-256/512/256	256	768	512	256	385	128

2.2 Variants of Collision Attacks

Let H, (m, m'), and IV be a hash function, a message pair, and an initial vector, respectively. A collision is a pair (m, m') satsfying $H(IV, m) = H(IV, m')$ where IV is set to 0 in `SPONGENT`. Let v be that is equal to the output of the previous block. A semi-free-start collision is a pair of (v, m) and (v, m') satisfying $H(v, m) = H(v, m')$ where $v \neq IV$.

2.3 Differential Characteristics

Differential cryptanalysis is one of the most powerful attacks on symmetric-key primitives proposed by Biham and Shamir [5]. In this attack, the attacker first tries to find a pair of input and output differences in which the input differences reach the output differences with a high probability and then mounts the key recovery on it. The time and data complexities depend on the ability to find an input-output pair difference with as high a probability as possible. Such search is the most important task in differential cryptanalysis. However, it is generally a hard task to find such pairs in practical time. Hence, we often use *differential characteristics* instead of calculating the probability of such pairs.

Let an r-round iterated block cipher be $E(\cdot) = f_r(\cdot) \circ \cdots \circ f_1(\cdot)$. Differential characteristic is defined as follows:

Definition 1 (Differential characteristic) *The differential characteristic is a sequence of differences over E defined as follows:*

$$\boldsymbol{D} = (\boldsymbol{d_0} \xrightarrow{f_1} \boldsymbol{d_1} \xrightarrow{f_2} \cdots \xrightarrow{f_r} \boldsymbol{d_r}) := (\boldsymbol{d_0}, \boldsymbol{d_1}, \cdots, \boldsymbol{d_r}),$$

where $\boldsymbol{d_i}$ is the difference at the output of i-th round for $0 < i \leq r$ and $\boldsymbol{d_0}$ is the difference at the input of the first round.

The Differential Characteristics Probability (DCP) is calculated by the product of the differential probability of each round on the well-known Markov cipher assumption [15] as follows:

$$DCP = \prod_{i=1}^{r} Prob.(\boldsymbol{d_{i-1}} \xrightarrow{f_i} \boldsymbol{d_i}).$$

For readability, we often use *weight* to express the probability of differential characteristics. A weight is defined as follows:

Definition 2 (Weight) *A weight w is a negated value of the binary logarithm of the probability P defined as follows:*

$$w = -\log_2 P$$

2.4 SAT-based Automatic Search Method

SAT Problems SAT problem is to solve whether a given Boolean formula can be made "True". A Boolean formula consists of only AND ($\wedge$), OR ($\vee$), and NOT ($\bar{\cdot}$) operations based on Boolean variables, and it can be converted into *Conjunctive Normal Form* (CNF). A CNF is constructed by the conjunction ($\wedge$) of the disjunction ($\vee$) on (possibly negated) Boolean variables, such as $\bigwedge_{a=0}^{i}(\bigvee_{b=0}^{j_a} x_{i,j})$, where $x_{i,j}$ is a Boolean variable. We call each disjunction $\bigvee_{b=0}^{j_a} c_{i,j}$ in a Boolean formula a *clause*. SAT solvers accept a CNF Boolean formula as their input.

Differential Characteristics Search by SAT. Recent studies demonstrated that the SAT-based automatic search method for identifying optimal differential characteristics can be much more efficient than the MILP- and CP-based ones [4, 29] for bit-oriented primitives. While the advantage is not clear in the case of byte-oriented ones, we opted for SAT and utilized Nicky and Bart's pure SAT-based method [23]. We leave for future research to test MILP- and CP-based techniques. Another possible research direction to improve our result is to integrate Sun et al.'s results into libraries like TAGADA, CASCADA, or CLAASP [3,16,26] to search for cipher vulnerabilities. In this case, rather than building a model directly, it is enough to represent the cipher in a specific format.

In the SAT-based search method for differential characteristics, we convert all differential propagation over primitives and its weight into a CNF. We call such a CNF a SAT model. Then, we give an objective function to identify the weight of the found differential characteristics, which is also expressed in a CNF, to a SAT model. Lastly, we give the created SAT model to a SAT solver. Hereafter, We briefly explain how to construct CNFs for each operation and the objective function.

Modeling for an XOR. Let $(a_0, a_1, \dots, a_{i-1})$ and b be the input and output differences of an XOR operation with i inputs, respectively, i.e., $a_0 \oplus a_1 \oplus \cdots \oplus a_{i-1} = b$. Let X be the set $\{(x_0, x_1, \dots, x_i) \in \mathbb{F}_2^{i+1} \mid (x_0 \oplus x_1 \oplus \cdots \oplus x_i) = 1\}$. The following variables and clauses are sufficient to express the differential propagation for an XOR with i inputs:

$$\mathcal{M}_{var} \leftarrow (a_0, a_1, \dots, a_{i-1}, b, x_0, x_1, \dots, x_i),$$
$$\mathcal{M}_{cla.xor} \leftarrow (a_0 \oplus x_0) \vee (a_1 \oplus x_1) \vee \cdots \vee (a_{i-1} \oplus x_{i-1}) \vee (b \oplus x_i),$$

where $\mathcal{M}_{var}$ and $\mathcal{M}_{cla.xor}$ denote the sets of Boolean variables and clauses, respectively.

Modeling for an XOR. Let (a_0, a_1) and b be the input and output differences of an XOR operation, respectively, i.e., $a_0 \oplus a_1 = b$. The following variables and clauses are sufficient to express the differential propagation for an XOR with i inputs:

$$\mathcal{M}_{var} \leftarrow (a_0, a_1, b),$$
$$\mathcal{M}_{cla.xor} \leftarrow \{(a_0 \vee a_1 \vee \bar{b}), (a_0 \vee \bar{a_1} \vee b), (\bar{a_0} \vee a_1 \vee b), (\bar{a_0} \vee \bar{a_1} \vee \bar{b}), \}$$

where $\mathcal{M}_{var}$ and $\mathcal{M}_{cla.xor}$ denote the sets of Boolean variables and clauses, respectively.

Modeling for an S-box. Let $\boldsymbol{a} = (a_0, a_1, \dots, a_{i-1})$ and $\boldsymbol{b} = (b_0, b_1, \dots, b_{i-1})$ be the input and output differences of an i-bit S-box, respectively. Additionally, we introduce additional Boolean variables $\boldsymbol{p} = (p_0, p_1, \dots, p_{j-1})$, where j is the maximum weight of the differential propagation through an i-bit S-box and $p_q \in \{0, 1\}$ for $1 \leq q \leq j-1$, to count the differential probability as an S-box is a probabilistic operation. With these Boolean variables, we construct the following Boolean formula:

$$f(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{p}) = \begin{cases} 1 & \textit{if } \Pr(\boldsymbol{a} \rightarrow \boldsymbol{b}) = 2^{-\sum_{q=0}^{j-1} p_q}, \\ 0 & \textit{otherwise.} \end{cases}$$

Then, we extract a set A, which contains all vectors satisfying $f(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) = 0$ as follows:

$$A = \{(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \in \mathbb{F}_2^{2i+j} \mid f(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) = 0\},$$

where $\boldsymbol{x} = (x_0, x_1, \ldots, x_{i-1})$, $\boldsymbol{y} = (y_0, y_1, \ldots, y_{i-1})$, and $\boldsymbol{z} = (z_0, z_1, \ldots, z_{j-1})$. Since A is a set of invalid patterns in a model of an S-box, we ban these patterns by the following clauses:

$$\bigvee_{c=0}^{i-1}(a_c \oplus x_c) \vee \bigvee_{d=0}^{i-1}(b_d \oplus y_d) \vee \bigvee_{e=0}^{j-1}(p_e \oplus z_e) = 1,\ (\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \in A. \tag{1}$$

The remaining vectors, identical to $\overline{A}$, are a set of valid patterns. Therefore, these clauses extract the differential propagation with corresponding weight over an i-bit S-box. Note that the solution space of $|A|$ clauses about $(\boldsymbol{a},\boldsymbol{b},\boldsymbol{p})$ in Eq. (1) is identical to that of the following Boolean function:

$$g(\boldsymbol{a},\boldsymbol{b},\boldsymbol{p}) = \bigwedge_{(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})\in\mathbb{F}_2^{2i+j}} \left(g(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \vee \bigvee_{c=0}^{i-1}(a_c \oplus x_c^{\eta}) \vee \bigvee_{d=0}^{i-1}(b_d \oplus y_d^{\eta}) \vee \bigvee_{e=0}^{j-1}(p_e \oplus z_e^{\eta}) \right).$$

This equation is called the *product-of-sum* of g. We know that we can reduce the number of clauses in g by Quine-McCluskey algorithm [21, 24, 25] and Espresso algorithm [9], shown in [18, 28, 29]. We use Espresso logic minimizer[4] to reduce clauses in g. Therefore, the following variables and clauses are sufficient to express the differential propagation with corresponding weight over an i-bit S-box:

$$\mathcal{M}_{var} \leftarrow (a_0, a_1, \ldots, a_{i-1}, b_0, b_1, \ldots, b_{i-1}, p_0, p_1, \ldots, p_{j-1}),$$
$$\mathcal{M}_{cla.sbox} \leftarrow min\left(g(\boldsymbol{a},\boldsymbol{b},\boldsymbol{p})\right),$$

where $\mathcal{M}_{cla.xor}$ denotes the set of clauses.

For the evaluation to count the number of active S-boxes, we must consider whether an S-box is active instead of its weight. Therefore, we can implement it by replacing variables $\boldsymbol{p}$ to a, indicating whether an S-box is active. We represent it as $\mathcal{M}_{cla.sbox^*}$.

Objective Function. To identify the weight of the found differential characteristics, we set the objective function that suppresses the sum of all variables $(p_0, p_1, \ldots, p_{n-1})$ expressing weight by a specific value k as follows:

$$\sum_{i=0}^{n-1} p_i \leq k.$$

[4] https://ptolemy.berkeley.edu/projects/embedded/pubs/downloads/espresso/index.htm

Eq.2 is called *Boolean cardinality constraints*. According to Erlacher et al.'s work [10], `kmtotalizer` [20] is a good choice to implement it. Therefore, the following variables and clauses are sufficient to express the objective function:

$$(\mathcal{M}_{cla.obj(k)}, \mathcal{M}_{var}) \leftarrow \texttt{kmtotalizer}(\Sigma_{i=0}^{n-1} p_i \leq k),$$

where $\mathcal{M}_{cla.obj(k)}$ denotes the set of clauses.

Modeling for an Entire SAT model. We construct the whole SAT model $\mathcal{SAT}_{model}$ to identify the differential characteristics with the weight less than or equal to k with the above clauses and Boolean variables as follows:

$$\mathcal{SAT}_{model} \leftarrow (\mathcal{M}_{var}, \mathcal{M}_{cla.xor}, \mathcal{M}_{cla.sbox}, \mathcal{M}_{cla.obj(k)}).$$

Then, we give $\mathcal{SAT}_{model}$ to a SAT solver and solve it. If a SAT solver returns "SAT", we obtain the differential characteristic with the weight less than or equal to k. Otherwise, we increase k and repeat this procedure until a SAT solver returns "SAT". To count the number of active S-boxes, we can use the same model replaced $\mathcal{M}_{cla.sbox}$ to $\mathcal{M}_{cla.sbox^*}$ with the same procedure. It should be mentioned that this SAT model contains only CNFs for an XOR, S-box, and objective function because our target `SPONGENT` has only these operations.

Modeling of Difference and Value Transitions Unlike block ciphers, the underlying permutation in permutation-based primitives does not involve round keys. Therefore, considering the dependence of differential transitions over multiple rounds, the risk of the actual state and differential characteristics becoming incompatible increases. In most MILP or SAT-based models for searching differential characteristics, only differential transitions are considered, and they are handled independently across different rounds, which may result in finding invalid differential characteristics for the actual state.

Liu et al [17] designed a model that considers both differential transitions and value transitions in differential characteristics search. They independently construct models to describe differential transitions and value transitions, and connect them using a model that describes the differential value relationships in nonlinear operations.

2.5 Previous Results and Obstacles

The designers have performed a collision resistance evaluation using rebound attacks [22] on 6-round `SPONGENT`-88/80/8 [7]. However, the computational complexity is $2^{55.2}$, which is much larger than the claimed security of 2^{40}, making it an ineffective attack. Abdelraheem showed differential and linear distinguisher for the underlying permutations [1]. Zhang and Liu [32] demonstrated that an efficient distinguishing attack could be performed on all variations of `SPONGENT` permutations using a truncated differential attack with the meet-in-the-middle technique.

Despite these efforts, detailed analysis papers on collision attacks by third party are still lacking. To efficiently carry out collision attacks, bringing enough degrees of freedom is necessary, which is the number of value pairs satisfying the differential characteristics. For block-cipher-based hash functions, such as AES-like hashing, this can be conducted by counting the available DoF in each S-box independently, such as the super-S-box technique in the rebound attack [11].

In contrast, for sponge-construction-based hash functions, such as SPONGENT, finding value pairs satisfying the differential characteristics is challenging because their underlying permutations do not have a key-inserting operations. This imposes taking the dependency between values and differences in S-boxes over multiple rounds into account, making obtaining enough DoF quite complex. At CRYPTO 2020, Liu et al. showed that many differential characteristics in previous studies on sponge-based hash functions are invalid, i.e., there are no value pairs that satisfy the differential characteristics, and they presented an automatic search tool to find valid differential characteristics. [17].

In our method, we first compute value pairs in selected parts of the hash function, similar to the rebound attack. However, bit dependencies in the internal state must be considered for both active and inactive bits.

3 Evaluation of the Number of Value Pairs Satisfying Differential Characteristics

To address this problem for collision attacks on sponge-construction-based hash functions, we propose a new method to efficiently evaluate the available degrees of freedom that satisfy a given set of differential characteristics for key-less permutations. The core idea of our method is to categorize the input bits by leveraging the dependency model of values and differences introduced by Liu et al. [17]. The proposed method consists of five steps, outlined as follows:

Step 1: Modeling by Liu et al.'s Method [17]. We describe the relationship between value transitions and difference transitions in the product of terms. To express these value-and-difference transitions, we adopt the method proposed by Liu et al. [17]. Specifically, to model these transitions, we construct a truth table that captures the value-and-difference transitions and utilize the Espresso tool to derive a corresponding Boolean function.

Step 2: Assignning a Given Differential Characteristic. We assign a given differential characteristic to the variables expressing the differences. Consequently, the product of terms exclusively consists of variables that represent value transitions satisfying a given differential characteristic.

Step 3: Expressing All Internal Variables by Input Variables. After creating the product of terms whose variables express a value transition with a given differential characteristic, we convert all internal-value variables into the input-value variables that can be easily conducted by analyzing the *Algebraic Normal Form* (ANF) of S-boxes in the case of SPONGENT. Then, we attempt to reduce the number of terms in the product of terms. This can be realized by simplify-logic tools. In this paper, we use simplify-logic() in SymPy library for Python.

Step 4: Grouping Input Variables. We analyze all terms in the product of terms and grouping the input variables depending on how they depend on each other, which is a core step of Grouping method.

Step 5: Calculating Total Available DoF. We count the number of value pairs for each group independently, which can be realized by solving the SAT model multiple times for each group. The obtained results are equivalent to the available DoF in each group. Therefore, the total available DoF is calculated by the product of the available DoF in each group.

Because Step 1 can be realized as a simple application of Liu et al.'s method [17] and Step 2 is just a processs of the allocation for variables regarding a given differential characteristic, we will focus on elaborating on the detailed procedure of the Grouping method in Step 3, 4, and 5 in the next section. The conceptual diagram of the Grouping method is shown in Fig. 3.

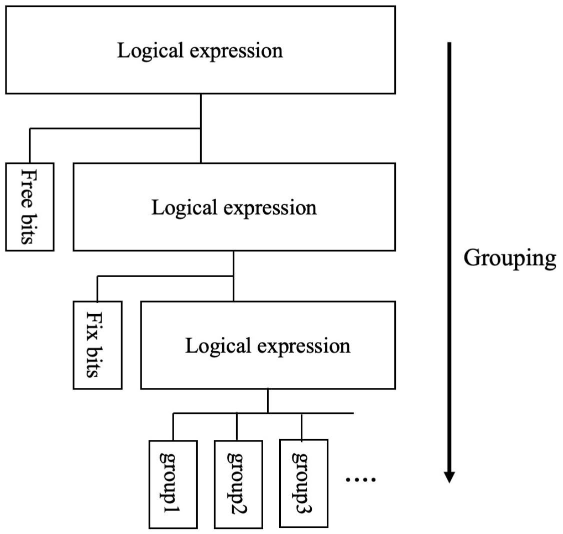

Fig. 3: Overview diagram of Grouping method

3.1 Grouping Method to Collect Value Pairs

Expressing All Internal Variables by Input Variables. After Step 1, and 2, the product of terms only consists of the variables expressing the internal values. Since the available DoF is brought from the message and the chaining

Algorithm 1: Grouping of input-value variables.

```
Data: $\mathcal{C}$
Result: $\mathcal{G}$
1 $a = 1$
2 for $i = 1$ to $\alpha$ do
3     for $j = 0$ to $l - 1$ do
4         if $v_i$ in $\mathcal{C}^j$ and $v_i$ is not in any group then
5             All literals in $\mathcal{C}^j$ are grouped to the $a$-th group
6             $a \leftarrow a + 1$
7         else if $v_i$ in $\mathcal{C}^j$ and $v_i$ is already in $x$-th group then
8             All literals in $\mathcal{C}^j$ are grouped to the $x$-th group
```

value in semi-free-start collision setting, we need to express the product of terms only by these states, which are called the input values in this paper. This can be easily conducted by analyzing ANF of the S-box in the case of SPONGENT. This procedure allows us to investigate the dependency of the input values, which is necessary to efficiently estimate the available DoF in the later steps.

Grouping Input Variables and Enumerating Available DoF in Each Group. To verify a given value-and-difference transition model expressed in the product of terms, all terms must be "True". Therefore, we categorize variables, which express the bits of the message and chaining value, into groups depending on which terms they are in. Then, we find all (or enough) assignments of literals in each group independently by solving the SAT model multiple times, equivalent to the number of valid value pairs with a given differential characteristic.

To find assignments of literals in a particular group, we first try to find one assignment for all literals. Then, we fix all literals to the found assignment except literals in a particular group in which we want to find all assignments of variables. This allows us to find all assignment by solving SAT models multiple times that can be efficiently carried out as this procedure can be viewed as an *incremental SAT problem.*

In the Grouping Method, dependencies between the internal values and differences can be categorized by "input values," which is essential to ensure valid differential characteristics and collect value pairs. Consequently, it is imperative to analyze it from the initial round and analyzing numerous rounds using the Grouping Method is challenging due to the computational cost.

Let $\mathcal{C} = \{C^0, C^1, \ldots, C^{l-1}\}$ and $C^i = \{c^i_0, c^i_1, \ldots, c^i_{m_i-1}\}$ be the Bool function describing value-and-difference transition and the term constructing $\mathcal{C}$ where c^i_j and m_i denote literals constructing C^i and the number of literals in C^i, respectively. Algorithm 1 shows how to categorize all input-value variables. In Algorithm 1, we assume to collect the value pairs satisfying a given differential characteristics regarding the function with the α-bit input, i.e., $c^i_j \in \{v_1, v_2, \ldots, v_\alpha\}$ where $0 \leq i < l, 0 \leq j < m_i$.

If a group has only one literal, we call this input-value variable as a *fixed bit* because it must be fixed to 0 or 1. If an input-value variable is not in any groups, we call it as a *free bit* because it can be either 0 or 1 not depending on other input-value variables.

We take the 2 rounds of SPONGENT-160/160/80 as an example. Table 3 shows the differential characteristic in this example. We show the product of sums describing the value-and-difference transition regarding Table 3 as follows:

$$
\begin{aligned}
& v_{49} \bigwedge \neg v_{52} \bigwedge v_{57} \bigwedge \neg v_{60} \bigwedge v_{67} \bigwedge v_{75} \bigwedge (v_{65}|v_{68}) \bigwedge (v_{73}|v_{76}) \bigwedge (v_{50}|\neg v_{51}) \\
& \bigwedge (v_{51}|\neg v_{50}) \bigwedge (v_{58}|\neg v_{59}) \bigwedge (v_{59}|\neg v_{58}) \bigwedge (v_{66}|\neg v_{65}) \bigwedge (v_{74}|\neg v_{73}) \bigwedge (\neg v_{66}|\neg v_{68}) \\
& \bigwedge (\neg v_{74}|\neg v_{76}) \bigwedge (v_{78} \oplus v_{79} \oplus (v_{77} \wedge v_{80}) \oplus (v_{78} \wedge v_{79} \wedge v_{80})) \\
& \bigwedge (v_{61} \oplus (v_{61} \wedge v_{64}) \oplus (v_{62} \wedge v_{63}) \oplus (v_{62} \wedge v_{64}) \oplus (v_{63} \wedge v_{64}) \oplus (v_{62} \wedge v_{63} \wedge v_{64})) \\
& \bigwedge (v_{77} \oplus (v_{77} \wedge v_{80}) \oplus (v_{78} \wedge v_{79}) \oplus (v_{78} \wedge v_{80}) \oplus (v_{79} \wedge v_{80}) \oplus (v_{78} \wedge v_{79} \wedge v_{80})) \\
& \bigwedge \neg (v_{53} \oplus (v_{53} \wedge v_{56}) \oplus (v_{54} \wedge v_{55}) \oplus (v_{54} \wedge v_{56}) \oplus (v_{55} \wedge v_{56}) \oplus (v_{54} \wedge v_{55} \wedge v_{56})) \\
& \bigwedge \neg (v_{69} \oplus (v_{69} \wedge v_{72}) \oplus (v_{70} \wedge v_{71}) \oplus (v_{70} \wedge v_{72}) \oplus (v_{71} \wedge v_{72}) \oplus (v_{70} \wedge v_{71} \wedge v_{72})) \\
& \bigwedge ((v_{58} \wedge v_{59})|\neg (v_{50} \wedge v_{51})|(v_{61} \oplus (v_{61} \wedge v_{64}) \oplus (v_{62} \wedge v_{63}) \oplus (v_{62} \wedge v_{64}) \\
& \oplus (v_{63} \wedge v_{64}) \oplus (v_{62} \wedge v_{63} \wedge v_{64}))) \bigwedge ((v_{74} \oplus (v_{73} \wedge v_{76}) \oplus (v_{74} \wedge v_{76})) \\
& |\neg (v_{66} \oplus (v_{65} \wedge v_{68}) \oplus (v_{66} \wedge v_{68}))|(v_{78} \oplus v_{79} \oplus (v_{77} \wedge v_{80}) \oplus (v_{78} \wedge v_{79} \wedge v_{80}))) \\
& \bigwedge ((v_{65} \oplus v_{66} \oplus v_{68} \oplus (v_{65} \wedge v_{68}))|\neg (v_{73} \oplus v_{74} \oplus v_{76} \oplus (v_{73} \wedge v_{76})) \\
& |(v_{77} \oplus (v_{77} \wedge v_{80}) \oplus (v_{78} \wedge v_{79}) \oplus (v_{78} \wedge v_{80}) \oplus (v_{79} \wedge v_{80}) \oplus (v_{78} \wedge v_{79} \wedge v_{80}))) \\
& \bigwedge \neg (v_{70} \oplus v_{71} \oplus (v_{69} \wedge v_{72}) \oplus (v_{70} \wedge v_{71} \wedge v_{72}))
\end{aligned}
$$

,

where v_i denotes the input-value variable from the leftmost bit.

Table 3: The differential characteristic of the 2-round SPONGENT-160/160/80. Differences are displayed in hexadecimal.

Rounds	Differences at the input of each round
1(Input)	0000000000006060b0b00000000000000000000000000000000000000000
2	000000000000000000550000000000000050000000000000000000000000
3 (Output)	0000000000000000000000000000000000c0004000000000000000000000

Table 4: Grouping results of the 2-round SPONGENT-160/160/80.

Category	Bits	Values
Free bits	48	v_{25}, v_{13}, v_{28}, v_{46}, v_{23}, v_{22}, v_{19}, v_{37}, v_{17}, v_{11}, v_{30}, v_{8}, v_{41}, v_{6}, v_{32}, v_{43}, v_{45}, v_{16}, v_{21}, v_{47}, v_{31}, v_{14}, v_{34}, v_{36}, v_{18}, v_{33}, v_{4}, v_{38}, v_{48}, v_{29}, v_{5}, v_{26}, v_{3}, v_{20}, v_{27}, v_{1}, v_{39}, v_{15}, v_{40}, v_{12}, v_{24}, v_{44}, v_{42}, v_{9}, v_{35}, v_{7}, v_{2}, v_{10}
Fixed bits	6	v_{49}, $\neg v_{52}$, v_{57}, $\neg v_{60}$, v_{67}, v_{75}
group1	4	v_{53}, v_{56}, v_{54}, v_{55}
group2	4	v_{69}, v_{72}, v_{70}, v_{71}
group3	8	v_{59}, v_{50}, v_{64}, v_{63}, v_{51}, v_{58}, v_{61}, v_{62}
group4	10	v_{80}, v_{78}, v_{77}, v_{73}, v_{68}, v_{66}, v_{65}, v_{76}, v_{79}, v_{74}

After applying Algorithm 1, the input-value variables are grouped as in Table 4. Since the input-value variables in the same group depends on each other and input-value variables in another group are independent, we can enumerate the assignment of the input-value variables group by group.

Calculating Total Available DoF. After Grouping the input-value variables, we attempt to find value pairs satisfying a given differential characteristic by solving the SAT model multiple times. We first try to find one solution. If there are no solutions, a given differential characteristic is invalid. Otherwise, we then count the number of value pairs satisfying differential characteristic by solving the SAT model multiple times. For the input-value variables categorized in free bits, they can take both 0 or 1. For the input-value variables categorized in groups, we count the number of solutions by solving the SAT models multiple times. Let a, N_f, and N_x be the number of groups, the number of free bits, and the number of solutions in the x-th group, respectively. Since value pairs in each group can be calculated independently, the total number of value pairs satisfying a given differential characteristic can be calculated as follows:

$$2^{N_f} \times N_1 \times N_2 \times \cdots \times N_a.$$

which is equivalent to the available DoF.

3.2 Application to Collision Attacks

We apply the Grouping method to collision attacks by collecting enough number of the value pairs satisfying a part of differential characteristics and finding a collision pair satsfying the remaining part of differential characteristic as follows.

- Firstly, we divide a hash function H into an upper part H_u and a lower part H_b such that $H = H_b \circ H_u$. Then, we search the differential characteritics for H, which can occur collisions, taking only the weight in H_b into consideration.

– Next, we model the value-and-difference transitions with SAT by Liu et al.'s method [17] so as to confirm the validity of the differential characteristic in H_b. If there are no valid value pairs in H_b, we search the differential characteristic for H again.
– Then, we apply the Grouping method to H_u and obtain the enough number of DoFs to satisfy the differential characteristics in H_b. Let N_{DoF} and p be the available DoF in H_u and the probability of the differential characteristics found in the previous step, respectively. If $N_{DoF} \geq p^{-1}$, we can theoretically obtain $N_{DoF} \cdot p$ pairs satisfying the differential characteristic for H, meaning to find the colliding pairs.

Remarks. Above procedure only ensures the validity for the differential characteristics in H_u and H_b independently, i.e., the validity of the entire differential characteristic for H is not guaranteed. Therefore, we support the validity of the entire differential characteristic by showing the existence of numerous number of the value pairs satisfying the differential characteristics in H_u. Moreover, we show practical collision pairs for several variants of `SPONGENT` in Sect. 4 to enhance our assumption.

Besides, we always apply the Grouping method from the 1st round in all attacks because dependencies between the internal values and differences can be categorized by "input values," which is essential to ensure valid differential characteristics and collect value pairs in the Grouping method. Consequently, it is imperative to analyze it from the initial round, and analyzing numerous rounds using the Grouping method is challenging due to the computational cost. This is why we consistently apply the Grouping method from the first round. Introducing the Grouping method in a middle round is intriguing and remains as future work.

4 Applications to `SPONGENT`

In this section, we apply the grouping method to `SPONGENT` to construct a collision attack. First, we describe the method for searching differential characteristics suitable for the collision attack. Then, we apply the Grouping method to the identified differential characteristics to conduct the collision attacks to each variant of `SPONGENT`.

Overview. We first evaluate the optimal differential characteristics that can be used to find collisions. Then, we try to carry out the collision attacks with the procedure described in Sect. 3.2. For the number of rounds in H_u, we set it to at most 5 rounds because it can be too complex to carry out the Grouping method to more than 5 rounds regarding `SPONGENT`. Besides, we always apply the Grouping method from the first round since the distribution of differences at the middle rounds is complex, making the Grouping method quite expensive.

4.1 Searching Differential Characteristics for Collisions

For the variants with the parameters of $(r = n)$, we consider two cases:

1. Differences in the rate part after one block call is zero
2. Differences in the capacity part after one block call is zero and the difference in the second message will cancel out the differences in the internal state.

For the variants with the parameters $(r \neq n)$, we consider the case where differences in the capacity part after one block call is zero and the difference in the second message will cancel out the differences in the internal state. The results of the DCP evaluation for the collision attack are shown in Table 5.

Table 5: Summary of DCP for collision attacks on each variant, where - means that it was computationally infeasible.

Rounds	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
`SPONGENT`-88/80/8	2	7	-	-	54	58	60	61	68	72	85	89	97	107	107	118	119
`SPONGENT`-88/176/88	2	4	8	12	20	29	43	54	63	67	73	86	-	-	-	-	-
`SPONGENT`-128/128/8	2	-	-	-	-	93	86	88	90	92	96	107	-	-	-	-	-
`SPONGENT`-128/256/128	2	4	8	12	20	28	40	48	52	56	62	66	76	82	92	103	-
`SPONGENT`-160/160/16	2	6	-	-	-	69	51	75	85	84	80	88	-	-	-	-	-
`SPONGENT`-160/160/80	2	48	12	20	28	34	39	45	50	56	61	67	72	78	83	89	-
`SPONGENT`-160/320/160	2	4	8	12	20	28	41	52	61	71	-	-	-	-	-	-	-
`SPONGENT`-224/224/16	2	6	-	-	-	-	-	96	100	106	-	-	-	-	-	-	-
`SPONGENT`-224/224/112	2	4	8	12	20	30	45	56	66	76	-	-	-	-	-	-	-
`SPONGENT`-224/448/224	2	4	8	12	20	28	44	54	73	-	-	-	-	-	-	-	-
`SPONGENT`-256/256/16	2	6	-	-	-	-	-	81	85	87	102	120	-	-	-	-	-
`SPONGENT`-256/256/128	2	4	8	12	20	28	41	49	56	60	64	70	78	87	97	104	110
`SPONGENT`-256/512/256	2	4	8	12	20	28	40	48	52	56	62	68	76	86	94	-	-

4.2 Collision Attacks on the `SPONGENT`

We show the detailed procedure of collision attacks on several variants of `SPONGENT`. In addition, the procedure to find semi-free-start collisions on `SPONGENT-128/256/128` and `SPONGENT-160/160/80` are shown in the Appendix B. Due to the page limitation, the remaining results of (semi-free-start) collisions on other variants of `SPONGENT` are summarized in Appendix A. The implementation codes for the collision attacks on `SPONGENT` are available on https://anonymous.4open.science/r/SPONGENT-Collision-83F7/README.md.

Attacks on 11-round `SPONGENT`-160/160/80

Step 1. Divide the round function into the first 2 rounds and the remaining 9 rounds such that $H = H_{(3\to11)} \circ H_{(1\to2)}$. Evaluate the differential characteristic for H taking the weight in $H_{(3\to11)}$, which must be less than the weight of the claimed security, into consideration. As a result, we obtain the differential characteristics shown in Table 6 whose probability in $H_{(3\to11)}$ is 2^{-61}.

Step 2. Construct the model for the value-and-difference transitions for $H_{(3\to11)}$ by Liu et al.'s method [17], and verify that the differential characteristic is valid.

Step 3. Apply the Grouping method to $H_{(1\to2)}$. Table 7 shows the results after applying the Grouping method. There are 48 free bits and 6 fixed bits. The other input bits are categorized into 4 groups.

Step 4. Since the differential probability of $H_{(3\to11)}$ is 2^{-61}, at least 2^{61} DoF in $H_{(1\to2)}$ is needed to obtain one collision pair on average. As we can obtain 2^{48} DoF by free bits, we need to get 2^{13} DoF by 4 groups. To this end, we construct the value-and-difference transition model by SAT and solve the constructed SAT models multiple times. As a result, we found 2^3, 2^2, 2^5, and 2^4 in groups 1, 2, 3, and 4, respectively. Therefore, we obtain $2^{48} \times 2^3 \times 2^2 \times 2^5 \times 2^3 = 2^{61}$ DoF that is enough to find one collision pair.

Step 5. Finding the colliding pairs with 2^{61} in $H_{(1\to2)}$, which satisfy the differential characteristic in $H_{(3\to11)}$.

Complexity Analysis. The cost of Grouping input variables is negligible compared to those of enumerating the available DoF and finding a pair satisfying the differential characteristic in $H_{(3\to11)}$ Time complexities of enumerating the available DoF and finding a pair satisfying the differential characteristic in $H_{(3\to11)}$ are $\frac{2}{11} \times 2^{61}$ and $\frac{9}{11} \times 2^{61}$, respectively. Since these procedures can be independently conducted, the total time complexity is $\frac{2}{11} \times 2^{61} + \frac{9}{11} \times 2^{61} = 2^{61} < 2^{80}$.

Table 6: The differential characteristic of SPONGENT-160/160/80. Differences are displayed in hexadecimal.

Rounds	Differences at the input of each round	Prob.	Phase
1(Input)	0000000000006060b0b000000000000000000000		$H_{(1\to2)}$
2	0000000000000000005500000000000500000000		
3	00000000000000000000000c0004000000000000		$H_{(3\to11)}$
4	0000000000000000000000000044000004400000		
5	00000000000000000000000000c0006000c00060		
6	0000000008800440000000000000000000000000		
7	0066000000000000066000000000000000000000	2^{-61}	
8	c000600000000000000000000000000000000000		
9	1100000000000000000000000000000000000000		
10	3000000000000003000000000000000000000000		
11	0000000000000001008000000000000000000000		
Output	0008400000000000084000000000000000000000		

Table 7: Grouping results of `SPONGENT-160/160/80`. The leftmost bit is denoted as v_1. Time of $H_{(1\to2)}$ represents the lower bound of the computational complexity required for collecting DoF in $H_{(1\to2)}$.

Category	Bits	Values
Free bits	48	v_{25}, v_{13}, v_{28}, v_{46}, v_{23}, v_{22}, v_{19}, v_{37}, v_{17}, v_{11}, v_{30}, v_{8}, v_{41}, v_{6}, v_{32}, v_{43}, v_{45}, v_{16}, v_{21}, v_{47}, v_{31}, v_{14}, v_{34}, v_{36}, v_{18}, v_{33}, v_{4}, v_{38}, v_{48}, v_{29}, v_{5}, v_{26}, v_{3}, v_{20}, v_{27}, v_{1}, v_{39}, v_{15}, v_{40}, v_{12}, v_{24}, v_{44}, v_{42}, v_{9}, v_{35}, v_{7}, v_{2}, v_{10}
Fixed bits	6	v_{49}, $\neg v_{52}$, v_{57}, $\neg v_{60}$, v_{67}, v_{75}
group1	4	v_{53}, v_{56}, v_{54}, v_{55}
group2	4	v_{69}, v_{72}, v_{70}, v_{71}
group3	8	v_{59}, v_{50}, v_{64}, v_{63}, v_{51}, v_{58}, v_{61}, v_{62}
group4	10	v_{80}, v_{78}, v_{77}, v_{73}, v_{68}, v_{66}, v_{65}, v_{76}, v_{79}, v_{74}
Time of $H_{(1\to2)}$		$2^{58.5}$

Attacks on 15-round `SPONGENT`-256/256/128

Step 1. Divide the round function into the first 2 rounds and the remaining 9 rounds such that $H = H_{(5\to15)} \circ H_{(1\to4)}$. Evaluate the differential characteristic for H taking the weight in $H_{(5\to15)}$, which must be less than the weight of the claimed security, into consideration. As a result, we obtain the differential characteristics shown in Table 8 whose probability in $H_{(5\to15)}$ is 2^{-79}.

Step 2. Construct the model for the value-and-difference transitions for $H_{(5\to15)}$ by Liu et al.'s method [17], and verify that the differential characteristic is valid.

Step 3. Apply the Grouping method to $H_{(1\to4)}$. Table 9 shows the results after applying the Grouping method. There are 64 free bits and 12 fixed bits. The other input bits are categorized into 2 groups.

Step 4. Since the differential probability of $H_{(5\to15)}$ is 2^{-79}, at least 2^{79} DoF in $H_{(1\to4)}$ is needed to obtain one collision pair on average. As we can obtain 2^{64} DoF by free bits, we need to get 2^{15} DoF by 4 groups. To this end, we construct the value-and-difference transition model by SAT and solve the constructed SAT models multiple times. As a result, we found 2^{12}, and 2^{13} in groups 1 and 2 respectively. Therefore, we obtain $2^{64} \times 2^{12} \times 2^{13} = 2^{79}$ DoF that is enough to find one collision pair.

Step 5. Finding the colliding pairs with 2^{79} in $H_{(1\to4)}$, which satisfy the differential characteristic in $H_{(5\to15)}$.

Complexity Analysis.

The cost of Grouping input variables is negligible compared to those of enumerating the available DoF and finding a pair satisfying the differential characteristic in $H_{(5\to15)}$ Time complexities of enumerating the available DoF and finding a pair satisfying the differential characteristic in $H_{(5\to15)}$ are $\frac{4}{15} \times 2^{79}$ and $\frac{11}{15} \times 2^{79}$, respectively. Since these procedures can be independently conducted, the total time complexity is $\frac{4}{15} \times 2^{79} + \frac{11}{15} \times 2^{79} = 2^{79} < 2^{128}$.

Table 8: The differential characteristic of `SPONGENT-256/256/128`. Differences are displayed in hexadecimal. The data is shown in a folded form, with each state listed sequentially from left to right.

Rounds	Differences at the input of each round	Prob.	Phase
1(Input)	0000000000000000ee800000295000000000000000000000000000000000000000000000000000000000000000000000		$H_{(1\to4)}$
2	0000e0200000000000000000000000000000000000000000000e0e000000000000000000000000000000000000000000		
3	050000000000050000000000000000000000000000000000500000000000500000000000000000000000000000000000		
4	000000000000000000000000000000000000000000000002002000000002002000000000000000000000000000000000		
5	000000000000000000000000000000000009009000000000000000000000000000000000000000000000900900000000	2^{-79}	$H_{(5\to15)}$
6	000000000000000000000000000000000000000000000000000000000000000000000900000000000900000000000000		
7	000000000000000000000000000000000000000020020000000000000000000000000000000000000000000000000000		
8	000000000021000000000000000000000000000000000000000000000210000000000000000000000000000000000000		
9	00c00000000000c000000000000000000000000000000000c00000000000c00000000000000000000000000000000000		
10	400400000000400400000000000000000000000000000000000000000000000000000000000000000000000000000000		
11	000000000000000000000000000000000000000000000009009000000000000000000000900900000000000000000000		
12	000000000000000000000000000000000000000000000000000000000000000000000000000000000000900000900000		
13	000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000140		
14	000000000000000000000060000000000000000000000060000000000000000000000000000000000000000000000000		
15	000080000080000000000000000000000000000000000000000000000000000000000000000000000000000000000000		
Output	028000000000000000000000280000000000000000000000000000000000000000000000000000000000000000000000		

Table 9: Grouping results of `SPONGENT-256/256/128`. The leftmost bit is denoted as v_1. Time of $H_{(1\to4)}$ represents the lower bound of the computational complexity required for collecting DoF in $H_{(1\to4)}$.

Category	Bits	Values
Free bits	64	v_{13}, v_{23}, v_{12}, v_{43}, v_{53}, v_{60}, v_{3}, v_{34}, v_{37}, v_{14}, v_{11}, v_{30}, v_{19}, v_{49}, v_{4}, v_{25}, v_{59}, v_{64}, v_{18}, v_{22}, v_{8}, v_{24}, v_{51}, v_{58}, v_{21}, v_{16}, v_{56}, v_{17}, v_{39}, v_{33}, v_{1}, v_{15}, v_{32}, v_{28}, v_{55}, v_{62}, v_{52}, v_{6}, v_{50}, v_{42}, v_{54}, v_{10}, v_{38}, v_{48}, v_{45}, v_{46}, v_{26}, v_{7}, v_{47}, v_{40}, v_{44}, v_{31}, v_{35}, v_{27}, v_{29}, v_{2}, v_{63}, v_{5}, v_{57}, v_{36}, v_{9}, v_{20}, v_{41}, v_{61}
Fixed bits	12	v_{69}, v_{73}, v_{77}, v_{78}, $\neg v_{79}$, $\neg v_{101}$, $\neg v_{103}$, $\neg v_{104}$, $\neg v_{106}$, $\neg v_{107}$, v_{110}, $\neg v_{112}$
group1	16	v_{87}, v_{93}, v_{85}, v_{81}, v_{96}, v_{82}, v_{89}, v_{90}, v_{91}, v_{94}, v_{95}, v_{84}, v_{92}, v_{86}, v_{88}, v_{83}
group2	36	v_{100}, v_{65}, v_{67}, v_{68}, v_{105}, v_{75}, v_{115}, v_{123}, v_{66}, v_{119}, v_{114}, v_{74}, v_{99}, v_{109}, v_{124}, v_{71}, v_{120}, v_{122}, v_{127}, v_{108}, v_{102}, v_{111}, v_{116}, v_{128}, v_{72}, v_{80}, v_{117}, v_{97}, v_{113}, v_{118}, v_{125}, v_{70}, v_{76}, v_{121}, v_{98}, v_{126}
Time of $H_{(1\to4)}$		$2^{77.1}$

4.3 Practical Collision Pairs on Several Variants of `SPONGENT`

To demonstrate the validity of the proposed method, we show the actual collision pairs for several variants. We show the detailed procedure of collision attacks on `SPONGENT-88/176/88`. The collision pairs and their input pairs are shown in Appendix C due to the page limitation.

Attacks on 8-round `SPONGENT-88/176/88`

Step 1. Divide the round function into the first 4 rounds and the remaining 4 rounds such that $H = H_{(1\to4)} \circ H_{(5\to8)}$. Evaluate the differential characteristic for H taking the weight in $H_{(5\to8)}$, which must be less than the weight of the claimed security, into consideration. As a result, we obtain the differential characteristics shown in Table 10 whose probability in $H_{(5\to 8)}$ is 2^{-28}.

Step 2. Construct the model for the value-and-difference transitions for $H_{(5\to 8)}$ by Liu et al.'s method [17], and verify that the differential characteristic is valid.

Step 3. Apply the Grouping method to $H_{(1\to 4)}$. Table 11 shows the results after applying the Grouping method. There are 32 free bits and 6 fixed bits. The other input bits are categorized into 1 group.

Step 4. Since the differential probability of $H_{(5\to 8)}$ is 2^{-28}, at least 2^{28} DoF in $H_{(1\to 4)}$ is needed to obtain one collision pair on average. As we can obtain 2^{32} DoF by free bits. To this end, we construct the value-and-difference transition model by SAT and solve constructed SAT models, and we found more than one state. Therefore, we obtain $2^{32} \times 1 = 2^{32}$ DoF that is enough to find one collision pair.

Step 5. Finding the colliding pairs with 2^{32} in $H_{(1\to 4)}$, which satisfy the differential characteristic in $H_{(5\to 8)}$.

Complexity Analysis. The cost of Grouping input variables is negligible compared to those of enumerating the available DoF and finding a pair satisfying the differential characteristic in $H_{(5\to 8)}$ Time complexities of enumerating the available DoF and finding a pair satisfying the differential characteristic in $H_{(5\to 8)}$ are $\frac{4}{8} \times 2^{28}$ and $\frac{4}{8} \times 2^{28}$, respectively. Since these procedures can be independently conducted, the total time complexity is $\frac{4}{8} \times 2^{28} + \frac{4}{8} \times 2^{28} = 2^{28} < 2^{43}$.

Table 10: The differential characteristic of SPONGENT-88/176/88. Differences are displayed in hexadecimal.

Rounds	Differences at the input of each round	Prob.	Phase
1(Input)	000000000d89b0000c6fa000000000000000000000000000000000000000000000		
2	006060000000000000000000000000000000000000000000000070700000000000		$H_{(1\to 4)}$
3	000000000000050005000000000000000000000000000000000000000000000000		
4	000000000000000000000000000000000000220000000000000000000000000000		
5	0000000000000000000000000c00000000000000000000000000000000c0000000		
6	000000200000000000000000000000000000000000000000000000000000000010	2^{-28}	$H_{(5\to 8)}$
7	000000000000000010100000000000000000000000000000010100000000000000		
8	000050000000a000000000000000000000000050000000a0000000000000000000		
Output	000000000000000000000000000000000010000000200000000040000000800000		

Table 11: Grouping results of `SPONGENT-88/176/88`. The leftmost bit is denoted as v_1. Time of $H_{(1\to4)}$ represents the lower bound of the computational complexity required for collecting DoF in $H_{(1\to4)}$.

Category	Bits	Values
Free bits	32	$v_{19}, v_{26}, v_1, v_5, v_{15}, v_{12}, v_{23}, v_3, v_{20}, v_{17}, v_8, v_{18}, v_{30}, v_{31}, v_7, v_{16}, v_{13}, v_{25}, v_4, v_{29}, v_2, v_6, v_{22}, v_{11}, v_{24}, v_9, v_{14}, v_{10}, v_{21}, v_{27}, v_{28}, v_{32}$
Fixed bits	6	$v_{38}, v_{51}, v_{69}, \neg v_{70}, v_{81}, \neg v_{83}$
group1	50	$v_{82}, v_{57}, v_{55}, v_{85}, v_{78}, v_{74}, v_{50}, v_{88}, v_{52}, v_{39}, v_{53}, v_{67}, v_{79}, v_{63}, v_{60}, v_{87}, v_{84}, v_{75}, v_{73}, v_{76}, v_{37}, v_{47}, v_{40}, v_{48}, v_{56}, v_{34}, v_{42}, v_{54}, v_{62}, v_{49}, v_{58}, v_{65}, v_{64}, v_{41}, v_{86}, v_{35}, v_{45}, v_{61}, v_{59}, v_{71}, v_{44}, v_{66}, v_{46}, v_{80}, v_{36}, v_{33}, v_{77}, v_{43}, v_{72}, v_{68}$
Time of $H_{(1\to4)}$		$2^{27.0}$

5 Conclusion

In this paper, we proposed a novel method to efficiently identify value pairs that satisfy differential characteristics for keyless permutations. To address the existing problem, we introduced a grouping method that classifies input variables into categories such as free bits, fixed bits, and interdependent groups. We applied this method to variants of `SPONGENT` and, for the first time, demonstrated collision and semi-free-start collision attacks on reduced variants. It should be noted that the results of our attacks do not immediately affect the security of Spongent as there are still enough security margin.

References

1. Abdelraheem, M.A.: Estimating the probabilities of low-weight differential and linear approximations on present-like ciphers. In: Kwon, T., Lee, M., Kwon, D. (eds.) Information Security and Cryptology - ICISC 2012 - 15th International Conference, Seoul, Korea, November 28-30, 2012, Revised Selected Papers. Lecture Notes in Computer Science, vol. 7839, pp. 368–382. Springer (2012). https://doi.org/10.1007/978-3-642-37682-5_26, https://doi.org/10.1007/978-3-642-37682-5_26
2. Aumasson, J., Henzen, L., Meier, W., Naya-Plasencia, M.: Quark: A lightweight hash. J. Cryptol. **26**(2), 313–339 (2013). https://doi.org/10.1007/S00145-012-9125-6, https://doi.org/10.1007/s00145-012-9125-6
3. Bellini, E., Gérault, D., Grados, J., Huang, Y.J., Makarim, R.H., Rachidi, M., Tiwari, S.K.: CLAASP: A cryptographic library for the automated analysis of symmetric primitives. In: Carlet, C., Mandal, K., Rijmen, V. (eds.) Selected Areas in Cryptography - SAC 2023 - 30th International Conference, Fredericton, Canada, August 14-18, 2023, Revised Selected Papers. Lecture Notes in Computer Science, vol. 14201, pp. 387–408. Springer (2023). https://doi.org/10.1007/978-3-031-53368-6_19, https://doi.org/10.1007/978-3-031-53368-6_19

4. Bellini, E., Piccoli, A.D., Formenti, M., Gerault, D., Huynh, P., Pelizzola, S., Polese, S., Visconti, A.: Differential cryptanalysis with sat, smt, milp, and cp: a detailed comparison for bit-oriented primitives. Cryptology ePrint Archive, Paper 2024/105 (2024)
5. Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems. Journal of CRYPTOLOGY **4**(1), 3–72 (1991)
6. Bogdanov, A., Knezevic, M., Leander, G., Toz, D., Varici, K., Verbauwhede, I.: spongent: A lightweight hash function. In: Preneel, B., Takagi, T. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6917, pp. 312–325. Springer (2011). https://doi.org/10.1007/978-3-642-23951-9_21, https://doi.org/10.1007/978-3-642-23951-9_21
7. Bogdanov, A., Knezevic, M., Leander, G., Toz, D., Varici, K., Verbauwhede, I.: SPONGENT: the design space of lightweight cryptographic hashing. IEEE Trans. Computers **62**(10), 2041–2053 (2013). https://doi.org/10.1109/TC.2012.196, https://doi.org/10.1109/TC.2012.196
8. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: an ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings. Lecture Notes in Computer Science, vol. 4727, pp. 450–466. Springer (2007). https://doi.org/10.1007/978-3-540-74735-2_31, https://doi.org/10.1007/978-3-540-74735-2_31
9. Brayton, R.K., Hachtel, G.D., McMullen, C.T., Sangiovanni-Vincentelli, A.L.: Logic Minimization Algorithms for VLSI Synthesis, The Kluwer International Series in Engineering and Computer Science, vol. 2. Springer (1984)
10. Erlacher, J., Mendel, F., Eichlseder, M.: Bounds for the Security of Ascon against Differential and Linear Cryptanalysis. IACR Trans. Symmetric Cryptol. **2022**(1), 64–87 (2022)
11. Gilbert, H., Peyrin, T.: Super-sbox cryptanalysis: Improved attacks for aes-like permutations. In: FSE. Lecture Notes in Computer Science, vol. 6147, pp. 365–383. Springer (2010)
12. Guo, J., Peyrin, T., Poschmann, A.: The PHOTON family of lightweight hash functions. In: Rogaway, P. (ed.) Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6841, pp. 222–239. Springer (2011). https://doi.org/10.1007/978-3-642-22792-9_13, https://doi.org/10.1007/978-3-642-22792-9_13
13. International Organization for Standardization: Iso/iec 29192-2:2019 - information security – lightweight cryptography – part 2: Block ciphers (02 2019), https://www.iso.org/standard/67173.html, available online: https://www.iso.org/standard/67173.html
14. Jean, J., Naya-Plasencia, M., Peyrin, T.: Improved rebound attack on the finalist grøstl. In: Canteaut, A. (ed.) Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers. Lecture Notes in Computer Science, vol. 7549, pp. 110–126. Springer (2012). https://doi.org/10.1007/978-3-642-34047-5_7, https://doi.org/10.1007/978-3-642-34047-5_7
15. Lai, X., Massey, J.L., Murphy, S.: Markov ciphers and differential cryptanalysis. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 547, pp. 17–38. Springer (1991)

16. Libralesso, L., Delobel, F., Lafourcade, P., Solnon, C.: Automatic generation of declarative models for differential cryptanalysis. In: Michel, L.D. (ed.) 27th International Conference on Principles and Practice of Constraint Programming, CP 2021, Montpellier, France (Virtual Conference), October 25-29, 2021. LIPIcs, vol. 210, pp. 40:1–40:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021). https://doi.org/10.4230/LIPICS.CP.2021.40, https://doi.org/10.4230/LIPIcs.CP.2021.40
17. Liu, F., Isobe, T., Meier, W.: Automatic verification of differential characteristics: Application to reduced gimli. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III. Lecture Notes in Computer Science, vol. 12172, pp. 219–248. Springer (2020). https://doi.org/10.1007/978-3-030-56877-1_8, https://doi.org/10.1007/978-3-030-56877-1_8
18. Liu, Y., Wang, Q., Rijmen, V.: Automatic search of linear trails in ARX with applications to SPECK and chaskey. In: ACNS. Lecture Notes in Computer Science, vol. 9696, pp. 485–499. Springer (2016)
19. Lu, X., Li, B., Liu, M., Lin, D.: Improved conditional differential attacks on lightweight hash family QUARK. Cybersecur. **5**(1), 12 (2022). https://doi.org/10.1186/S42400-021-00108-3, https://doi.org/10.1186/s42400-021-00108-3
20. Martins, R., Joshi, S., Manquinho, V.M., Lynce, I.: Incremental Cardinality Constraints for MaxSAT. CoRR **abs/1408.4628** (2014)
21. McCluskey, E.J.: Minimization of Boolean functions. The Bell System Technical Journal **35**(6), 1417–1444 (1956)
22. Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl. In: FSE. Lecture Notes in Computer Science, vol. 5665, pp. 260–276. Springer (2009)
23. Mouha, N., Preneel, B.: A proof that the ARX cipher salsa20 is secure against differential cryptanalysis. IACR Cryptol. ePrint Arch. p. 328 (2013)
24. Quine, W.V.: The problem of simplifying truth functions. The American mathematical monthly **59**(8), 521–531 (1952)
25. Quine, W.V.: A way to simplify truth functions. The American mathematical monthly **62**(9), 627–631 (1955)
26. Ranea, A., Rijmen, V.: Characteristic automated search of cryptographic algorithms for distinguishing attacks (cascada). IET Information Security **16**(6), 470–481 (2022). https://doi.org/https://doi.org/10.1049/ise2.12077, https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/ise2.12077
27. Shiba, R., Sakamoto, K., Liu, F., Minematsu, K., Isobe, T.: Integral and impossible-differential attacks on the reduced-round lesamnta-lw-bc. IET Inf. Secur. **16**(2), 75–85 (2022). https://doi.org/10.1049/ISE2.12044, https://doi.org/10.1049/ise2.12044
28. Sun, L., Wang, W., Wang, M.: More accurate differential properties of LED64 and midori64. IACR Trans. Symmetric Cryptol. **2018**(3), 93–123 (2018)
29. Sun, L., Wang, W., Wang, M.: Accelerating the Search of Differential and Linear Characteristics with the SAT Method. IACR Trans. Symmetric Cryptol. **2021**(1), 269–315 (2021)
30. Wang, Q., Grassi, L., Rechberger, C.: Zero-sum partitions of PHOTON permutations. In: Smart, N.P. (ed.) Topics in Cryptology - CT-RSA 2018 - The Cryptographers' Track at the RSA Conference 2018, San Francisco, CA, USA, April 16-20, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10808, pp. 279–299. Springer (2018). https://doi.org/10.1007/978-3-319-76953-0_15, https://doi.org/10.1007/978-3-319-76953-0_15

31. Zhang, G., Liu, M.: A distinguisher on present-like permutations with application to SPONGENT. IACR Cryptol. ePrint Arch. p. 236 (2016), http://eprint.iacr.org/2016/236
32. Zhang, G., Liu, M.: A distinguisher on present-like permutations with application to SPONGENT. Sci. China Inf. Sci. **60**(7), 72101 (2017). https://doi.org/10.1007/S11432-016-0165-6, https://doi.org/10.1007/s11432-016-0165-6

A Summary of Attacks on Other Variants of SPONGENT

In this section, we show the results of (semi-free start) collision attacks on the other variants of SPONGENT. Table 12 shows the summary of the results of our attacks. Due to the page limitations, we omit the detailed procedures of several our attacks. The procedures of all attacks are the same as described in Sect. 4.

Table 12: Summary of collision attack results. Rounds marked with * indicate actual collision pairs found. †: invalid attack as it is less efficient than generic attack

	Collision		Semi-free-start		Full	Collision
	Rounds	Time	Rounds	Time	round	security
SPONGENT-88/80/8	-	-	8*	2^{17}	45	2^{40}
SPONGENT-88/176/88	9 8*	2^{42} 2^{28}	9*	2^{33}	135	2^{44}
SPONGENT-128/128/8	-	-	8*	2^{22}	70	2^{64}
SPONGENT-128/256/128	12 8*	2^{58} 2^{28}	14	2^{63}	195	2^{64}
SPONGENT-160/160/16	-	-	7	2^{47}	90	2^{80}
SPONGENT-160/160/80	11	2^{61}	14	2^{78}	120	2^{80}
SPONGENT-160/320/160	11 8*	2^{56} 2^{28}	13	2^{79}	240	2^{80}
SPONGENT-224/224/16	-	-	8	2^{91}	120	2^{112}
SPONGENT-224/224/112	10	2^{60}	11	2^{62}	170	2^{112}
SPONGENT-224/448/224	10 8*	2^{47} 2^{21}	11	2^{57}	340	2^{112}
SPONGENT-256/256/16	-	-	11	2^{110}	140	2^{128}
SPONGENT-256/256/128	15	2^{79}	15	2^{75}	195	2^{128}
SPONGENT-256/512/256	15 9*	2^{73} 2^{28}	15	2^{73}	385	2^{128}

B Applications of Semi-free-start Collisions

B.1 Attacks on 14-round SPONGENT-128/256/128

Attack Procedure

Step 1. Divide the round function into the first 2 rounds and the remaining 9 rounds such that $H = H_{(5\to14)} \circ H_{(1\to4)}$. Evaluate the differential characteristic for H taking the weight in $H_{(5\to14)}$, which must be less than the weight of the claimed security, into consideration. As a result, we obtain the differential characteristics shown in Table 13 whose probability in $H_{(5\rightarrow14)}$ is 2^{-63}.

Step 2. Construct the model for the value-and-difference transitions for $H_{(5\rightarrow14)}$ by Liu et al.'s method [17], and verify that the differential characteristic is valid.

Step 3. Apply the Grouping method to $H_{(1\rightarrow4)}$. Table 14 shows the results after applying the Grouping method. There are 4 free bits and 33 fixed bits. The other input bits are categorized into 3 groups.

Step 4. Since the differential probability of $H_{(5\rightarrow14)}$ is 2^{-63}, at least 2^{63} DoF in $H_{(1\rightarrow4)}$ is needed to obtain one collision pair on average. As we can obtain 2^4 DoF by free bits, we need to get 2^{59} DoF by 4 groups. To this end, we construct the value-and-difference transition model by SAT and solve the constructed SAT models multiple times. As a result, we found 2^{20}, 2^{20}, 2^{29} in groups 1, 2, and 3 respectively. Therefore, we obtain $2^4 \times 2^{20} \times 2^{20} \times 2^{19} = 2^{63}$ DoF that is enough to find one collision pair.

Step 5. Finding the colliding pairs with 2^{63} in $H_{(1\rightarrow4)}$, which satisfy the differential characteristic in $H_{(5\rightarrow14)}$.

Complexity Analysis. The cost of grouping input variables is negligible compared to those of enumerating the available DoF and finding a pair satisfying the differential characteristic in $H_{(5\rightarrow14)}$ Time complexities of enumerating the available DoF and finding a pair satisfying the differential characteristic in $H_{(5\rightarrow14)}$ are $\frac{4}{14} \times 2^{63}$ and $\frac{10}{14} \times 2^{63}$, respectively. Since these procedures can be independently conducted, the total time complexity is $\frac{4}{14} \times 2^{63} + \frac{10}{14} \times 2^{63} = 2^{63} < 2^{128}$.

Table 13: The differential characteristic of SPONGENT-128/256/128 for semi-free-start collisions.

Rounds	Differences at the input of each round	Prob.	Phase
1(Input)	9dba9526599000003990e330037000000000000000000000000000000000000000000000000000000000000000000000		$H_{(1\rightarrow4)}$
2	cc000000000000000000000000077600000000000000000077700000000000000000000000000000000000000000000000		
3	300000070000700000000000000000000000000000000000000000000000000000000000000000000000000000000000		
4	000000000000000000000000180100000000000000000000000000000000000000000000000000000000000000000000		
5	000000b0000000000000000000000000000000000000000000000000000b00000000000000000000000000000000000000000	2^{-63}	$H_{(5\rightarrow14)}$
6	040000000000040000000000000000000000000000000000000000000000000000000000000000000000000000000000		
7	000000000000000000000000000000000000000000000000200200000000000000000000200200000000000000000000		
8	000000000000000000000000000000000000900000900000000000000000000000000000000000000000900000900000		
9	000000000000000000000000000000000000000000000000000000000000000000000000000000000140000000000140		
10	000000000000000000006006000000000000000000006006000000000000000000000000000000000000000000000000		
11	000090000090000000000000000000000000000000000000000000000000000000000000000000000000000000000000		
12	000000000000000000000000000000000000000000000000000000000000000000000000028000000000000000000000		
13	000000000000000000600000000000000000000000000000000000000000000000600000000000000000000000000000		
14	000040000000000000000000000000000000000000000000000000000000000040000000000000000000000000000000		
Output	000000000000000000000000000000000000000000000000100000000000000010000000010000000000000010000000		

Table 14: Grouping results of SPONGENT-128/256/128 (semi-free-start). The left-most bit is denoted as v_1. Time of $H_{(1\to4)}$ represents the lower bound of the computational complexity required for collecting DoF in $H_{(1\to4)}$.

Category	Bits	Values
Free bits	4	$v_{85}, v_{89}, v_{65}, v_{101}$
Fix bits	33	$v_2, \neg v_3, v_6, \neg v_{11}, v_{15}, \neg v_{13}, \neg v_{18}, \neg v_{19}, v_{22}, \neg v_{24}, \neg v_{25}, \neg v_{27}, \neg v_{28}, v_{34}, \neg v_{36}, \neg v_{38}, \neg v_{39}, \neg v_{42}, \neg v_{43}, v_{68}, \neg v_{67}, v_{70}, v_{71}, v_{74}, v_{75}, \neg v_{81}, v_{88}, \neg v_{87}, v_{92}, \neg v_{91}, v_{104}, \neg v_{103}, \neg v_{108}$
group1	64	$v_{172}, v_{132}, v_{163}, v_{140}, v_{187}, v_{162}, v_{159}, v_{155}, v_{179}, v_{185}, v_{157}, v_{176}, v_{165}, v_{149}, v_{186}, v_{150}, v_{183}, v_{169}, v_{166}, v_{152}, v_{175}, v_{133}, v_{168}, v_{171}, v_{147}, v_{144}, v_{154}, v_{181}, v_{164}, v_{170}, v_{135}, v_{138}, v_{174}, v_{178}, v_{190}, v_{151}, v_{161}, v_{158}, v_{130}, v_{184}, v_{156}, v_{131}, v_{191}, v_{173}, v_{137}, v_{142}, v_{180}, v_{134}, v_{192}, v_{182}, v_{160}, v_{177}, v_{167}, v_{136}, v_{188}, v_{139}, v_{146}, v_{148}, v_{143}, v_{145}, v_{153}, v_{141}, v_{189}, v_{129}$
group2	64	$v_{199}, v_{252}, v_{197}, v_{204}, v_{236}, v_{249}, v_{194}, v_{256}, v_{217}, v_{193}, v_{241}, v_{216}, v_{220}, v_{205}, v_{244}, v_{198}, v_{206}, v_{245}, v_{214}, v_{221}, v_{196}, v_{225}, v_{247}, v_{232}, v_{218}, v_{211}, v_{237}, v_{235}, v_{212}, v_{255}, v_{223}, v_{208}, v_{200}, v_{203}, v_{201}, v_{250}, v_{246}, v_{229}, v_{251}, v_{233}, v_{254}, v_{243}, v_{195}, v_{222}, v_{215}, v_{202}, v_{207}, v_{248}, v_{228}, v_{210}, v_{253}, v_{238}, v_{234}, v_{242}, v_{213}, v_{240}, v_{230}, v_{227}, v_{209}, v_{226}, v_{231}, v_{239}, v_{219}, v_{224}$
group3	219	$v_{279}, v_{260}, v_{60}, v_{31}, v_{10}, v_{370}, v_{78}, v_{100}, v_{119}, v_{295}, v_{267}, v_{53}, v_{300}, v_{96}, v_{61}, v_{304}, v_{272}, v_{265}, v_5, v_{90}, v_{259}, v_{283}, v_{297}, v_{326}, v_{51}, v_{312}, v_{26}, v_{356}, v_{35}, v_{372}, v_1, v_{41}, v_{76}, v_{348}, v_{357}, v_{340}, v_{328}, v_{266}, v_{371}, v_{57}, v_{378}, v_{375}, v_{55}, v_{365}, v_{368}, v_{287}, v_{257}, v_{123}, v_{301}, v_{84}, v_{284}, v_{341}, v_{360}, v_{20}, v_{319}, v_{44}, v_{63}, v_{107}, v_{369}, v_{285}, v_{327}, v_{110}, v_{308}, v_{269}, v_{21}, v_{268}, v_{350}, v_{324}, v_{331}, v_{353}, v_{99}, v_{258}, v_{344}, v_{111}, v_9, v_{296}, v_{80}, v_{128}, v_{77}, v_{307}, v_{336}, v_{298}, v_{325}, v_{321}, v_{50}, v_{73}, v_{349}, v_{310}, v_{333}, v_{12}, v_{105}, v_{17}, v_{359}, v_{280}, v_{282}, v_{37}, v_{367}, v_{126}, v_{30}, v_{289}, v_{347}, v_{292}, v_{95}, v_7, v_{122}, v_{98}, v_{261}, v_{47}, v_{276}, v_{278}, v_{45}, v_{115}, v_{106}, v_{262}, v_{293}, v_{313}, v_{62}, v_{97}, v_{270}, v_{315}, v_{291}, v_{116}, v_{381}, v_{361}, v_{317}, v_{314}, v_{127}, v_{329}, v_{380}, v_{383}, v_{355}, v_{377}, v_{29}, v_{338}, v_{311}, v_{303}, v_{306}, v_{16}, v_{286}, v_{121}, v_{273}, v_{332}, v_{274}, v_{364}, v_{352}, v_{343}, v_{376}, v_{305}, v_{345}, v_{83}, v_8, v_{363}, v_{366}, v_{263}, v_{125}, v_{281}, v_{290}, v_{112}, v_{46}, v_{49}, v_{66}, v_{120}, v_{316}, v_{48}, v_{358}, v_{384}, v_{79}, v_{373}, v_{117}, v_{102}, v_{342}, v_{58}, v_{323}, v_{64}, v_{351}, v_{322}, v_{362}, v_{109}, v_{288}, v_{379}, v_{339}, v_{86}, v_{124}, v_{118}, v_{69}, v_{275}, v_{23}, v_{354}, v_{72}, v_{277}, v_{56}, v_{33}, v_{309}, v_{318}, v_4, v_{271}, v_{302}, v_{334}, v_{113}, v_{54}, v_{114}, v_{93}, v_{59}, v_{264}, v_{330}, v_{94}, v_{14}, v_{320}, v_{346}, v_{337}, v_{382}, v_{335}, v_{32}, v_{82}, v_{374}, v_{294}, v_{52}, v_{299}, v_{40}$
Time of $H_{(1\to4)}$		$2^{61.2}$

B.2 Attacks on 14-round SPONGENT-160/160/80

Attack Procedure

Step 1. Divide the round function into the first 2 rounds and the remaining 9 rounds such that $H = H_{(4\to14)} \circ H_{(1\to3)}$. Evaluate the differential characteristic for H taking the weight in $H_{(4\to14)}$, which must be less than the weight of the claimed security, into consideration. As a result, we obtain the differential characteristics shown in Table 15 whose probability in $H_{(4\to14)}$ is 2^{-78}.

Step 2. Construct the model for the value-and-difference transitions for $H_{(4\to14)}$ by Liu et al.'s method [17], and verify that the differential characteristic is valid.

Step 3. Apply the Grouping method to $H_{(1\to3)}$. Table 16 shows the results after applying the Grouping method. There are 160 free bits and 12 fixed bits. The other input bits are categorized into 3 groups.

Step 4. Since the differential probability of $H_{(4\to14)}$ is 2^{-78}, at least 2^{78} DoF in $H_{(1\to3)}$ is needed to obtain one collision pair on average. As we can obtain 2^{160} DoF by free bits if one or more states can be discovered from group 1. To this end, we construct the value-and-difference transition model by SAT and solve the constructed SAT model, and we found more than one state. Therefore, we obtain $2^{78} \times 1 = 2^{78}$ DoF that is enough to find one collision pair.

Step 5. Finding the colliding pairs with 2^{78} in $H_{(1\to3)}$, which satisfy the differential characteristic in $H_{(4\to14)}$.

Computational cost evaluation Ignoring the cost of Grouping, the computational cost required to find one collision pair is $\frac{3}{14} \times 2^{78} + \frac{11}{14} \times 2^{78} = 2^{78} < 2^{80}$(claimed security). Therefore, the attack is successful.

The cost of grouping input variables is negligible compared to those of enumerating the available DoF and finding a pair satisfying the differential characteristic in $H_{(4\to14)}$ Time complexities of enumerating the available DoF and finding a pair satisfying the differential characteristic in $H_{(4\to14)}$ are $\frac{3}{14} \times 2^{78}$ and $\frac{11}{14} \times 2^{78}$, respectively. Since these procedures can be independently conducted, the total time complexity is $\frac{3}{14} \times 2^{78} + \frac{11}{14} \times 2^{78} = 2^{78} < 2^{128}$.

Table 15: The differential characteristic of `SPONGENT-160/160/80` (semi-free-start). Differences are displayed in hexadecimal.

Rounds	Differences at the input of each round	Prob.	Phase
1(Input)	`a0ac6660fff0000000000000000000000000000000000000000000000000`		
2	`000000000000000000000000000000d70000000000000d07000000000000`		$H_{(1\to3)}$
3	`0000000000000000000000c000a000000000000000000000000000000000`		
4	`000004400000000000000000000000000000000000000000000000000000`		
5	`000000000000000000000000000000060000000000000060000000000000`		
6	`000000000000000000000080004000000000000000000000000000000000`		
7	`000004400000000000000440000000000000000000000000000000000000`		
8	`000000000000000000000000000000060003000000000060003000000000`		
9	`000000000000000000000088004400000000000000000000000000000000`	2^{-78}	$H_{(4\to14)}$
10	`00000cc0000000000000cc00000000000000000000000000000000000000`		
11	`000000000000000000000000000000000000000000000060003000000000`		
12	`000000000000000000000000004400000000000000000000000000000000`		
13	`000000000000000000000000000000000000c00000000000000c00000000`		
14	`000000000100800000000000000000000000000000000000000000000000`		
Output	`002100000000000000210000000000000000000000000000000000000000`		

Table 16: Grouping results of SPONGENT-160/160/80 (semi-free-start). The left-most bit is denoted as v_1. Time of $H_{(1\to3)}$ represents the lower bound of the computational complexity required for collecting DoF in $H_{(1\to4)}$.

Category	Bits	Values
Free bits	160	$v_{58}, v_{176}, v_{165}, v_{162}, v_{204}, v_{120}, v_{150}, v_{180}, v_{98}, v_{170}, v_{201}, v_{126}, v_{164}, v_{85}, v_{117}, v_{82}, v_{163}, v_{113}, v_{166}, v_{65}, v_{129}, v_{141}, v_{151}, v_{208}, v_{172}, v_{149}, v_{185}, v_{207}, v_{71}, v_{114}, v_{206}, v_{146}, v_{81}, v_{173}, v_{136}, v_{70}, v_{68}, v_{205}, v_{112}, v_{192}, v_{89}, v_{96}, v_{122}, v_{178}, v_{134}, v_{61}, v_{118}, v_{143}, v_{101}, v_{155}, v_{188}, v_{131}, v_{191}, v_{119}, v_{177}, v_{156}, v_{110}, v_{52}, v_{91}, v_{63}, v_{87}, v_{174}, v_{107}, v_{202}, v_{196}, v_{86}, v_{77}, v_{145}, v_{94}, v_{116}, v_{102}, v_{121}, v_{171}, v_{108}, v_{54}, v_{194}, v_{106}, v_{49}, v_{78}, v_{139}, v_{153}, v_{111}, v_{104}, v_{142}, v_{100}, v_{115}, v_{199}, v_{50}, v_{95}, v_{197}, v_{183}, v_{186}, v_{84}, v_{200}, v_{147}, v_{133}, v_{92}, v_{123}, v_{161}, v_{189}, v_{124}, v_{184}, v_{53}, v_{55}, v_{103}, v_{64}, v_{51}, v_{57}, v_{159}, v_{128}, v_{67}, v_{73}, v_{93}, v_{144}, v_{125}, v_{138}, v_{80}, v_{137}, v_{152}, v_{182}, v_{193}, v_{109}, v_{158}, v_{198}, v_{190}, v_{169}, v_{132}, v_{56}, v_{187}, v_{72}, v_{99}, v_{157}, v_{175}, v_{203}, v_{130}, v_{160}, v_{168}, v_{154}, v_{74}, v_{97}, v_{105}, v_{59}, v_{62}, v_{148}, v_{181}, v_{90}, v_{135}, v_{179}, v_{60}, v_{88}, v_{83}, v_{76}, v_{167}, v_{195}, v_{69}, v_{75}, v_{66}, v_{79}, v_{127}, v_{140}$
Fix bits	12	$v_1, \neg v_3, \neg v_{11}, \neg v_9, v_{14}, \neg v_{13}, v_{20}, \neg v_{17}, v_{24}, \neg v_{21}, v_{28}, \neg v_{25}$
group1	68	$v_8, v_{33}, v_{44}, v_{23}, v_{48}, v_{29}, v_{213}, v_{227}, v_{234}, v_4, v_{210}, v_{36}, v_{16}, v_{222}, v_7, v_{215}, v_{10}, v_{18}, v_{233}, v_{31}, v_{37}, v_{19}, v_{22}, v_{239}, v_6, v_{236}, v_{219}, v_{34}, v_{217}, v_{229}, v_{45}, v_{226}, v_5, v_{220}, v_{214}, v_{46}, v_{35}, v_{38}, v_{30}, v_{216}, v_{238}, v_{231}, v_{15}, v_{42}, v_{209}, v_{218}, v_{228}, v_{12}, v_{26}, v_{47}, v_{43}, v_{232}, v_2, v_{230}, v_{32}, v_{224}, v_{235}, v_{225}, v_{240}, v_{211}, v_{237}, v_{27}, v_{39}, v_{223}, v_{212}, v_{41}, v_{40}, v_{221}$
Time of $H_{(1\to3)}$	$2^{75.8}$	

C Values of Practical Collision Pairs

Collision pairs for several variants of SPONGENT are shown in Table 17.

Table 17: Practical Value. The binary sequence is interpreted starting from the leftmost bit and then converted to a hexadecimal representation. The asterisk (*) indicates that the capacity part of the output difference after one block becomes zero. A collision occurs if, in the second message input, a message is chosen such that the message difference in the rate part becomes zero.

	Semi-free-start	Rounds	Collision Initial State Pair	Output state pair after one block
SPONGENT-88/80/8*	√	8	99849783a23de4bef427d0	eac5019f85d14f1a00737a
			eb849783a23de4bef427d0	aac5019f85d14f1a00737a
SPONGENT-88/176/88		8	35fb4346c60adde3b1201b00000000000 000000000000000000000000000000000	7b3b46e4d322a1d2df8a76699c9006f38 2975d94c6c2721c044aef94b4b2412f83
			35fb4346cb836de3bd4fbb00000000000 000000000000000000000000000000000	7b3b46e4d322a1d2df8a76699c9006f38 2875d94c6e2721c044aaf94b4b2c12f83
SPONGENT-88/176/88	√	9	d7d70b10f19bd790f78b2dc55a0b036d2 54d9134af34fdbebe860cbcf86a5847d5	47e569326d361e19bfcf46156a3855a24 e6188f8846c540bbe30af0da266db4299
			6ed84110f14bd799422b2ac55a0b036d2 54d9134af34fdbebe860cbcf86a5847d5	47e569326d361e19bfcf46556a3855224 e6188f8856c540bbc30af09a666db4a91
SPONGENT-128/128/8*	√	7	c342f6693b4fb33ef56c4da6e1d9ce7d50	769f064e009136ef26462cb1beeeeafbd0
			4542f6693b4fb33ef56c4da6e1d9ce7d50	f59f064e009136ef26462cb1beeeeafbd0
SPONGENT-128/256/128		8	7170635ed5680d896650069ac6f1a8c70000000000000000 000000000000000000000000000000000000000000000000	b12cda2140dd3d1b4c2603576b93f5a86c725fef56dc1424 d756cd33f93934a629d79a69475674db62852870867ca2b9
			7170635ed5680d896650260ca34dafa70000000000000000 000000000000000000000000000000000000000000000000	b12cda2140dd3d1b4c2603576b93f5a86c725def76de1424 d576ef33d93b36a629d59a6b477676db42872a70867ea2bb
SPONGENT-160/320/160*		8	27ddc249cbcc5cef966bfb0a748106559bb59323000000000000000000000 000000000000000000000000000000000000000000000000000000000000	d79e0af164ad21793aada2f81d4c2f032bf75f312835c8743b1373f7c04d e409f24a352042a16b6a932155fbdf694aedca90ab79f7497f51ab29628e
			27ddc249cbcc5ecd900bf1f3748100259775932300000000000000000000 000000000000000000000000000000000000000000000000000000000000	879e0af574ad21793aada2f81d4c2f532bf75b212835c8743b1373f7c04d e409f24a352042a16b6a932155fbdf694aedca90ab79f7497f51ab29628e
SPONGENT-224/448/224		8	c3a2eaac434f7793ddd32253323311a37773225677 76ae483d4eba8d0000000000000000000000000000 000000000000000000000000000000000000000000 000000000000000000000000000000000000000000	a12e3217a8741ff53096e085ec41212f98127bfe26 e05ff82c6e06cd7a37c635eb83e44451b7f83c3a1d e1d0f25752d2546c4ff43852bed65003c66aa98b52 58fa133842a704d1940deb93a75d006cbfc741f8b5
			c3a2eaac434f11a4000477b667667794111477b322 23ae483d4eba8d0000000000000000000000000000 000000000000000000000000000000000000000000 000000000000000000000000000000000000000000	a12e3217a8741ff53096e085ec41212f98127bfe26 e05ff82c6e06cd7a37c63deb83e54451b7f83c3a1d e1d0f24752d0546c4ff43052bed75003c66aa98b52 58fa132842a504d1940deb93a75c006cbfc741f8b5
SPONGENT-256/512/256		9	3bb916a152a19bb360105c2bfa68296932f7ea3933868ba9 768292ca17c2d2d900000000000000000000000000000000 000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000	a3730378aaf6cc8dcbbb5a259d594688ab4926702cb7cf04 81a7d99fdfa24f8189b7f6534c52a45a50c9788d9b0d0d2b a20cad2fb189abb3fc2907670dbea12e3e78bb8b3e50b819 9ca5e449ebc876f9715f7dbf59aa14778e9cd3f710f8b59d
			3bb916a152a19bb360105c2bfa68296967f2ea3966838ba9 238792ca1717d2d900000000000000000000000000000000 000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000	a3730378aaf6cc8dcbbb5a259d594688ab4926702cb7cf04 81a7d99fdfa24f8189b7f6534c52a45a50e97a8d9b0d0d0b a20cad2fb189abb3fc2907670dbea12e3e78bb8b3e50b839 9ca5e449ebc876f9715f7dbf5baa14778e9cd1f710f8b5bd