CNF Characterization of Sets over $\mathbb{Z}_2^n$ and Its Applications in Cryptography

HU Xiaobo · XU Shengyuan · TU Yinzi · FENG Xiutao

Abstract In recent years, the automatic search has been widely used to search differential characteristics and linear approximations with high probability/correlation. Among these methods, the automatic search with the Boolean Satisfiability Problem (SAT, in short) gradually becomes a powerful cryptanalysis tool in symmetric ciphers. A key problem in the automatic search method is how to fully characterize a set $S \subseteq \{0,1\}^n$ with as few Conjunctive Normal Form (CNF, in short) clauses as possible, which is called a full CNF characterization (FCC, in short) problem. In this work, we establish a complete theory to solve a best solution of a FCC problem. Specifically, we start from plain sets, which can be characterized by exactly one clause. By exploring mergeable sets, we find a sufficient and necessary condition for characterizing a plain set. Based on the properties of plain sets, we further provide an algorithm for solving a FCC problem with S, which can generate all the minimal plain closures of S and produce a best FCC theoretically. We also apply our algorithm to S-boxes used in block ciphers to characterize their differential distribution tables and linear approximation tables. All of our FCCs are the best-known results at present.

Keywords Automatic cryptanalysis, Boolean Satisfiability Problem, Full CNF characterization

1 Introduction

Differential analysis [1] and linear analysis [2] are two of the most powerful techniques in cryptanalysis and play essential roles in the security evaluation of symmetric ciphers. The main idea of them is to construct a difference/linear trail with high probability/correlation. In recent years, it has become a new trend to search for differential and linear trails using automatic methods. In Eurocrypt 1994, Matsui [3] proposed an automatic search algorithm based on the branch and bound method. Since then, more and more automatic tools are introduced into

HU Xiaobo
Beijing Smartchip Microelectronics Technology Company Limited. Email: huxiaobo@sgchip.sgcc.com.cn
XU Shengyuan
Key Laboratory of Mathematics Mechanization, Academy of Mathematics and Systems Science, Chinese Academy of Sciences; University of Chinese Academy of Sciences. Email: xushengyuan18@amss.ac.cn
TU Yinzi
Beijing Smartchip Microelectronics Technology Company Limited. Email: tuyinzi@sgchip.sgcc.com.cn
FENG Xiutao
Key Laboratory of Mathematics Mechanization, Academy of Mathematics and Systems Science, Chinese Academy of Sciences. Email: fengxt@amss.ac.cn

*This research was supported by National Key Research and Development Project under Grant No. 2018YFA0704705 and CAS Project for Young Scientists in Basic Research (Grant No. YSBR-035).
◇

cryptanalysis, among which the methods based on Mixed Integer Linear Programming (MILP, in short) and Boolean Satisfiability Problem (SAT, in short) are the most widely used.

MILP is an optimization problem to find the maximum/minimum value of the objective function under linear constraints. There are many mature solvers available to solve this problem, such as Gurobi [4], CPLEX [5], and SCIP [6]. The MILP problem was first introduced into cryptanalysis in 2011 [7, 8] to search for the minimal number of active S-boxes at the word level for differential and linear trails. Later, Sun *et al.* [9] proposed a bit-level algorithm which enabled to search for differential and linear trails. In [10], Fu *et al.* searched differential and linear characteristics for ARX ciphers. By modelling the propagation of division property into an MILP problem, Xiang *et al.* [11] constructed integral distinguishers. In [12], Zero-correlation distinguishers were searched by MILP techniques. Sasaki *et al.* [13] found new impossible differential distinguishers with the help of MILP. Besides, the MILP method is also widely used in other cryptanalysis algorithms, such as boomerang attacks [14] and cube attacks [15].

SAT investigates whether there is a set of variable assignments for the given equations that makes the problem satisfiable. It is a deterministic problem and has been proven to be Non-deterministic Polynomial Complete (NPC, in short). There are also many mature and efficient solvers for SAT, such as Minisat [16], Cryptominisat [17], and CaDiCal [18]. The SAT-based method was applied to cryptanalysis in 2013 [19] for the first time. Later, it was used to search the optimal differential and linear trails of SIMON [20]. In ACNS 2016, Liu *et al.* [21] proposed an automatic analysis algorithm for ARX structured cryptography based on SAT. By establishing SAT models, more accurate differential probabilities of LED64 and Midori64 were obtained in FSE 2018 [22]. In addition to differential analysis and linear analysis, the SAT method can also be applied to other cryptanalysis algorithms, such as searching impossible differential trails [23] and integral distinguishers [24]. SAT method has demonstrated higher efficiency compared to MILP method in certain problems, and has achieved remarkable results in recent years in the field of cryptanalysis.

In comparison to Matsui's algorithm, automatic search algorithms based on MILP and SAT offer a higher level of automation, enhanced readability, and easier solvability. Therefore, the core problem of this kind of automatic search algorithm is how to build efficient models. For NPC problems, the dominant view is that the smaller the scale of the problem, the easier it is to solve. From a practical point of view, numerous experimental results show that runtimes of simpler models are generally shorter. Consequently, attackers usually tend to construct smaller scale models in practical. The scale of the model mainly depends on the number of variables and constraints (i.e., inequalities and clauses). The former parameter is fixed for the same cipher when no dummy variables are introduced. Based on this perspective, we thoroughly study the Conjunctive Normal Form (CNF, in short) characterization of cryptographic components in automatic cryptanalyses in this paper and aim to reduce the number of clauses in models and improve the efficiency of automatic cryptanalysis algorithms.

1.1 Related Work

The early characterization works based on MILP mostly constructed candidate inequalities by calling SAGEMath [25] to obtain convex hulls of sets and then using the greedy algorithm to select as few inequalities as possible [9]. In general, it is difficult to obtain enough high-quality inequalities by such a method, leading to a large number of inequalities to characterize the target set. Moreover, this method can not deal with the higher dimensional cases. In FSE 2020, Boura *et al.* [26] studied the algebraic structure of sets and presented properties of inequalities that can cut a special class of sets which is called the "ball". This kind of inequalities can be regarded as high-quality inequalities. By adding them to the candidate set and combining with solving the Set Covering Problem (SCP, in short), they got the full characterization with fewer inequalities. Subsequently, more and more works have focused on constructing high-quality inequalities by investigating the algebraic structure of sets [27, 28]. In [29], by studying the relationship between plain sets that can be characterized by an inequality and their characterization inequality, a sufficient and necessary condition for the plain set was proposed. Based on the properties of plain sets, an efficient algorithm was designed to construct a best full characterization with the minimal number of inequalities theoretically.

While full characterizations of MILP models has been thoroughly studied, such methods cannot be directly applied to characterizations of SAT models due to differences in the ranges of the coefficients. Although less theoretical research has been conducted on the CNF characterization method, the same framework can be applied, and there are some intrinsic connections still worth investigating. More specifically, Abdelkhalek *et al.* [30] proposed the characterization algorithm for large S-boxes based on the logical condition. Although their work focused on MILP models, its core was constructing a CNF characterization and transforming it into inequalities, making it essentially a characterization of SAT. First, the target of the characterization can be a differential distribution table, truncated differential distribution table, linear approximation table, and truncated linear approximation table, which are abbreviated as DDT, *-DDT, LAT, *-LAT respectively. For a given S-box, one of the above four objects is transformed into a truth table of a Boolean function, where the value of the possible pattern is 1, and the value of the impossible pattern is 0. After inputting this truth table into the software Logic Friday [31], a CNF characterization can be output by calling the Quine-McCluskey algorithm [32] and the Espresso algorithm [33] to simplify the Product-of-Sum representation. The Quine-McCluskey algorithm and the Espresso algorithm are two powerful used algorithms to find a simplified representation of a Boolean function in terms of a minimal number of literals or logic gates. They are also commonly used in the automated cryptanalysis for generating and reducing FCC. Although QM algorithm aims to generate all prime implicants, its efficiency is limited by the selection algorithm. Espresso algorithm, on the other hand, cannot guarantee optimality due to the use of heuristic strategies. Moreover, Boura *et al.* [26] also proposed a MILP characterization algorithm which is essentially CNF characterization. They searched all sets of the form $a \oplus prec(u)$ and selected a subset of them by solving the SCP. Indeed, Boura's algorithm was strongly related with the QM algorithm.

By applying the characterization theory described above, searches for optimal differential probabilities for LED64 and Midori64 were completed in [22]. Liu *et al.* [21] realized the optimal differential search for ARX cipher by modelling basic operations such as modular addition and Xor. Additionally, they used the sequence coding method to characterize the constraint $\sum_{i=0}^{n-1} x_i \leq k$. By introducing Matsui's conditions into previous search models and characterizing them with the sequence coding method, [34] found full rounds optimal differential/linear trails for many block ciphers. Later, Wang *et al.* [35] further improved the sequence encoding method to characterize Matsui's conditions and reduced the scale of the model by decreasing the number of variables and clauses. As a result, they found the best differential/linear trails for full round GIFT-128.

We notice that the traditional characterization method relies on the external software and can not guarantee a minimal number of clauses. Furthermore, the solving time is longer when the dimension of sets is relatively large, which results in the limitation of application scenarios. In order to address these challenges, our motivation is to propose a new algorithm to complete the full CNF characterization of a given set and achieve the provable optimality theoretically.

1.2 Our Contributions

For a given subset S of $\{0,1\}^n$, $\mathcal{C}$ is a set of CNF clauses such that the solution space of $\mathcal{C}$ on $\{0,1\}^n$ is S exactly. We call $\mathcal{C}$ a full CNF characterization (FCC, in short) of S. Our main contribution is to establish a complete theory to get a $\mathcal{C}$ with the minimal number of clauses.

Firstly, we focus on the plain set, which can be characterized by a clause. We present their essential properties in terms of clauses and sets respectively, and provide a sufficient and necessary condition for a plain set. Furthermore, we also explore the connections between linear integer inequalities and CNF clauses.

Secondly, based on the properties of plain sets, we propose an algorithm to construct a FCC for a given set. The new algorithm can obtain best FCCs for sets whose dimensions are up to 12. It is also efficient for sets with larger dimensions and can provide better results.

Finally, we apply our algorithm to various S-boxes used in block ciphers and get best FCCs for their DDTs and LATs as well as accelerate existing models, the results are listed in Table 2, Table 3 and Table 4. All of our results are the best-known at present.

1.3 Organization

This paper is organized as follows: Firstly necessary notations and some preliminaries are introduced in Section 2. In Section 3, the properties of the plain set which can be characterized by a CNF clause are studied and a necessary and sufficient condition for plain sets is proposed. In Section 4, an efficient algorithm to find a full CNF characterization with the minimal number of CNF clauses for a given set is introduced. In Section 5, we apply the new algorithm to characterize different S-boxes and get the best results so far.

2 Notations and Preliminaries

In this section we give a brief overview of some notations and definitions. Table 1 lists parts of notations.

2.1 Notations

Table 1: The notations used throughout the paper

Notation	Description
n	A positive integer
$\mathbb{Z}_2$	The set $\{0,1\}$
$\mathbb{Z}_2^n$	The set of all n-tuples over $\mathbb{Z}_2$, i.e., $\{0,1\}^n$
$\mathbb{Z}_*^n$	The set $\{0,1,*\}^n$
$\mathbb{Z}_n$	The integer set $\{0,1,\cdots,n-1\}$
$\mathbf{\Pi}_n$	The set of all subsets of $\mathbb{Z}_2^n$
$\mathbf{P}_n$	The set of all plain sets in $\mathbf{\Pi}_n$
x_i	The i-th bit of x
$wt(x)$	Hamming weight of x
e_i	An n-bit unit whose i-th element is 1 and others are 0
$x \oplus y$	Bitwise Xor between x and y
S	A subset of $\mathbb{Z}_2^n$
$\overline{S}$	The complementary set of S in $\mathbb{Z}_2^n$
$\mathcal{C}(S)$	The set of all the plain closures of S
$\wedge$, $\vee$	Logical operation And, Or
$\overline{x}$	Logical operation Not: $1 \oplus x$
$c : \bigvee_{i=0}^{n-1} a_i \vee (x_i \oplus c_i) = 1$	A CNF clause

2.2 Boolean Formula and CNF

Boolean formulas are formulas that consist of only boolean variables and the operators And ($\wedge$), Or ($\vee$), Not ($\bar{}$). SAT studies whether there is a set of boolean variables whose assignment satisfies the given boolean formulas. If so, the problem is said to be satisfiable; otherwise, it is said to be unsatisfiable. Since SAT is proven to be NPC, how to solve it efficiently is a hot research direction in recent years.

Usually, for a given boolean formula, researchers first represent it as the standard input format of SAT solver: the form of CNF, namely: $\bigwedge_{i=0}^{m-1} \left(\bigvee_{j=0}^{n_i-1} x_{i,j} \right) = 1$, where $x_{i,j}$ are boolean variables, constants or the negation of boolean variables. Each $\bigvee_{j=0}^{n_i-1} x_{i,j} = 1$ in a CNF is called a clause. The variables in a clause are joined by Or, that is, the single clause is true as long as one of its variables is true. Since different clauses are joined by And, the whole problem is satisfied if and only if each clause holds. To describe the form in which a boolean variable appears in a clause, we have the following definition:

Definition 2.1 Suppose $c : \bigvee_{i=0}^{n-1}(a_i \vee (x_i \oplus c_i)) = 1$, the coefficient of x_i in c is denoted as $c(x_i)$, where

$$c(x_i) := \begin{cases} 1, & \text{if } a_i = 0 \text{ and } c_i = 0, \\ -1, & \text{if } a_i = 0 \text{ and } c_i = 1, \\ 0, & \text{otherwise.} \end{cases}$$

When $c(x_i) = 1$, the variable x_i appears in a clause and we say its coefficient in this clause is 1; when $c(x_i) = -1$, the negation of the variable x_i appears and we say its coefficient is -1; otherwise, the coefficient is said to be 0, i.e., $c(x_i) = 0$.

Next we will explain how to characterize a cryptanalysis algorithm. Take the *-DDT of the S-box as a example, denote $f(x, y)$ as a Boolean function whose input is $2n$-bit, where $x = (x_0, x_1, \cdots, x_{n-1})$ and $y = (y_0, y_1, \cdots, y_{n-1})$ are input difference and output difference respectively. Set $f(x, y) = 1$ if and only if (x, y) is a possible pattern of the DDT, then the constraint condition $f(x, y) = 1$ is the full characterization of the *-DDT. The Product-of-Sum representation of f can be expressed as follows:

$$f(x,y) = \bigwedge_{c \in \{0,1\}^{2n}} \left(\alpha_c \vee \bigvee_{i=0}^{n-1} (x_i \oplus c_i) \vee \bigvee_{i=0}^{n-1} (y_i \oplus c_{n+i}) \right),$$

where $\alpha_c = f(c) \in \{0, 1\}$, $c = (c_0, c_1, \cdots, c_{2n-1})$. Then $f(x, y) = 1$ if and only if

$$\bigvee_{i=0}^{n-1} (x_i \oplus c_i) \bigvee_{i=0}^{n-1} (y_i \oplus c_{n+i}) = 1$$

holds for all c such that $\alpha_c = 0$. When at least one of the $(x_i \oplus c_i)$ and $(y_i \oplus c_{n+i})$ is 1, the result of the Or is 1, thus the solution space of $f(x, y) = 1$ is exactly $\{c \in \mathbb{Z}_2^n \mid \alpha_c = 1\}$, and the solution space of the above CNF is exactly all the possible patterns of the *-DDT when set $\alpha_c = 0$ for all impossible patterns. By using this method to characterize each component of the target cipher, the solution space of the whole model can be restricted to the solution space of the actual cryptanalysis problem. Thus the feasible solution obtained by the solver can be the differential/linear trails or distinguishers that attackers need.

In this paper, such characterization is known as the full CNF characterization which will be defined in the next section.

2.3 Full CNF Characterization of Sets over $\mathbb{Z}_2^n$

In automatic cryptanalysis, the core of all the modelling problems is characterizing a subset of $\mathbb{Z}_2^n$, that is, to build the corresponding CNF clause system whose solution set is exactly the target set. This problem is called the full CNF characterization problem of a finite set, and is defined as follows:

Definition 2.2 (Full CNF Characterization) If the solution set of a CNF system $\mathcal{C} : \bigwedge_{i=0}^{m-1} \left(\bigvee_{j=0}^{n_i-1} c_{i,j} \right) = 1$ is $S \in \mathbf{\Pi}_n$, then $\mathcal{C}$ is called the full CNF characterization of S. Elements in S are called feasible points of $\mathcal{C}$; elements in $\overline{S}$ are called infeasible points of $\mathcal{C}$.

We can always construct a FCC for a given set $S \in \mathbf{\Pi}_n$, that is to say, FCC always exists. Before presenting the formal proof, we make conventions on the meaning of the symbols c and $\mathcal{C}$ for convenience. Without confusion, we still use the notation c to denote the solution space of the CNF clause c, thus $c \in \mathbf{\Pi}_n$ and $\overline{c}$ means the complementary set of c in $\mathbb{Z}_2^n$ when considered as a set. Similarly, we have $\mathcal{C} = \bigcap_{i=0}^{m-1} c$ when $\mathcal{C}$ is viewed as a set. Next, we prove the existence of the FCC of S.

Theorem 2.1 *(Existence of the FCC) For an arbitrary given set $S \in \mathbf{\Pi}_n$, there must exist a FCC $\mathcal{C}$ of S.*

Proof Denote $\overline{S} = \{s^i | i = 0, \cdots, n-1\}$, we konw that $Z_2^n \setminus \{s^i\}$ is the solution space of the clause $c_i : \bigvee_{i=0}^{n-1}(x_i \oplus s_i^i) = 1$, where $s^i = (s_0^i, s_1^i, \cdots, s_{n-1}^i) \in \overline{S}$. Then $\mathcal{C} = \cap_{i=0}^{n-1} c_i = \cap_{i=0}^{n-1}(Z_2^n \setminus \{s_i\}) = Z_2^n \setminus (\cup_{i=0}^{n-1}\{s_i\}) = Z_2^n \setminus \overline{S} = S$, hence $\mathcal{C}$ is a FCC of S. □

A FCC is composed of multiple clauses and a set may have more than one FCC. To build more efficient models, researchers always concentrate on characterizations with fewer clauses. For a given set S, define the FCC of S with the minimal number of clauses as a best FCC of S.

First we mainly focus on the properties of a single clause which corresponding to a special class of subsets in $\mathbb{Z}_2^n$.

Definition 2.3 (Plain Set) For $S \in \mathbf{\Pi}_n$, we say S is plain if S can be characterized by a single clause. Denote the set of all the plain sets in $\mathbf{\Pi}_n$ as $\mathbf{P}_n$.

Obviously, $\varnothing, \mathbb{Z}_2^n \in \mathbf{P}_n$. We say $\varnothing$ and $\mathbb{Z}_2^n$ are trivial and $S \in \mathbf{P}_n \backslash \{\varnothing, \mathbb{Z}_2^n\}$ is non-trivial. Plain sets are a class of sets with many special properties which are important in constructions of FCCs. In more detail, each plain set corresponds to a CNF clause with respect to characterization. Conversely, the solution set of a CNF system is the intersection of some plain sets. Therefore, to design a general scheme for FCC construction, it is a key step to study the properties of plain sets.

2.4 Set Covering Problem

Suppose $\mathcal{U}$ is a given set, denote $\mathcal{S}$ as a subset of the power set of $\mathcal{U}$ which contains n elements whose union is $\mathcal{U}$. Set Covering Problem is aimed to find a minimal subset of $\mathcal{S}$ such that their union is equal to $\mathcal{U}$. The specific description is as follows:
Decision variables:

$$y_s = \begin{cases} 1, & \text{if } s \in \mathcal{S} \text{ is selected;} \\ 0, & \text{otherwise.} \end{cases}$$

Objective function:

$$\text{Minimize} \sum_{s \in \mathcal{S}} y_i.$$

Constraint conditions:

$$\sum_{s:e \in s} y_s \geq 1, \forall e \in \mathcal{U};$$

$$y_s \in \{0, 1\}, \forall s \in \mathcal{S}.$$

The objective function is set as the minimum value of the number of selected subsets and the constraint conditions ensure that all elements in $\mathcal{U}$ are covered.

SCP is a well-known NPC problem [36] which plays a crucial role in reducing the scale of models for better characterization in automatic cryptanalysis. Traditional MILP and SAT characterization can be divided into two steps: the first step is to generate sufficient characterization inequalities (clauses) for the given set and establish a full characterization as a candidate set; the second step is to remove redundant inequalities (clauses) from this candidate set, i.e., select the minimal number of inequalities (clauses) as the final full characterization. The second step can be completely converted into an SCP, where the set composed of all infeasible points is the target set $\mathcal{U}$, the set of points cut off by each candidate inequality (clause) is regarded as an element of $\mathcal{S}$, and all the candidate inequalities (clauses) can be regarded as $\mathcal{S}$. By selecting the minimal subset of $\mathcal{S}$ to cover $\mathcal{U}$, the minimal number of inequality (clause) characterization can be constructed. According to the definition, SCP is equivalent to a MILP which can be solved directly by MILP solvers and get the optimal value. For low dimensional sets, this method is efficient and can provide a theoretical guarantee of optimality.

However, for higher dimensional sets, the MILP model corresponding to the SCP can be complex due to a large number of candidate constraints and infeasible points, making it challenging to obtain the optimal solution directly. In such cases, alternative methods such as the greedy algorithm and heuristic algorithms are often adopted to get better solutions. For instance, many solvers that are more efficient than Gurobi have been proposed in the GECCO Competition, which can deal with higher dimensional cases. Among them, an effective framework [37, 38] can be described as follows: 1) The problem is simplifying by adopting reduction rules; 2) An initial feasible solution is generating by applying a PageRank-like constructive heuristic; 3) The SCP is transformed into a series of k-set covering decision subproblems; 4) Each subproblem is tackled with a fast local search procedure.

Despite the theoretical difficulty in proving optimality, a good solution for FCCs can still be obtained by solving the SCP for high dimensional sets. In this paper, we will utilize one of the aforementioned methods in a flexible manner to obtain a better solution.

3 Essential Properties of Plain Sets

In this section we will explore some essential properties of plain sets. Based on these properties, we further propose a sufficient and necessary condition for the plain set.

3.1 Plain Set and Degeneration

A FCC is composed of multiple CNF clauses combined by the $\wedge$ operations, that is, the boolean formula is satisfied if and only if every clause is true simultaneously. We first study the properties of a single CNF clause and its corresponding plain set. For a given CNF clause, the whole space $\mathbb{Z}_2^n$ can be divided into two categories according to whether this clause is satisfied: feasible points and infeasible points. Since boolean variables in the clause are concatenated by $\vee$, feasible points have the following property:

Property 3.1 Denote $c : \bigvee_{i\in I} x_i \vee \bigvee_{j\in J} \overline{x}_j = 1$, $x \in \mathbb{Z}_2^n$ is a feasible point of c if and only if there exists $i \in I$, s.t. $x_i = 1$ or $j \in J$, s.t. $x_j = 0$.

According to Property 3.1, the requirement for a point to be feasible of a clause is easy to satisfy: one of the components satisfying the specific value. Therefore, more information can be obtained by studying the properties of unfeasible point sets. Hence, conditions that need to be met for infeasible points of the same clause are considered. To start with, the characterization clause and its infeasible points have the following property:

Property 3.2 Suppose $S \in \mathbf{P}_n$ and c is a FCC of S, define three subsets as below:

$$I_0 := \{i | s_i = 0, \forall s \in \overline{S}\},$$

$$I_1 := \{i | s_i = 1, \forall s \in \overline{S}\},$$

$$I_2 := \{i | \exists s^0 \in \overline{S}, s.t.\ s_i^0 = 1 \text{ and } \exists s^1 \in \overline{S}, s.t.\ s_i^1 = 0\}.$$

When $i \in I_0$, it can be concluded that $c(x_i) \neq -1$, i.e., $\overline{x}_i$ will not appear in c; when $i \in I_1$, then $c(x_i) \neq 1$, i.e., x_i will not appear in c; when $i \in I_2$, then $c(x_i) = 0$, i.e., both x_i and $\overline{x}_i$ will not appear in c.

Proof The first two statements obviously hold according to Property 3.1, then we will prove the third statement by contradiction. Since c is a FCC of the plain set S, when $c(x_i) = 1$, all s that satisfy $s_i = 1$ are preserved by c; when $c(x_i) = -1$, then s that satisfy $s_i = 0$ are preserved by c. Since $I_2 \neq \varnothing$, then $c \cap \overline{S} \neq \varnothing$, which is contradicted with the definition of FCC. □

The above property reflects the relation between coefficients of clauses and their infeasible sets. From the point of view of the clause, when $c(x_i) = 0$, we say the clause degenerates at the i-th dimension. In this case, x_i does not influence the characterization, and the space of the set can be reduced from $\mathbb{Z}_2^n$ to $\mathbb{Z}_2^{n-1}$. According to Property 3.2, when $I_2 \neq \varnothing$, the characterization clause of S must correspond to degenerations of i-th dimension for all $i \in I_2$. Thus, infeasible points are very closely related to the degeneration. In non-degenerate cases, every x_i appears, that is, $c(x_i) \neq 0$, $i = 0, \cdots, n-1$, by using $x_i^{'} := \overline{x}_i$ to instead x_i, all such clauses are eventually transformed into the same form, which is called the trivial clause:

Definition 3.1 (Trivial Clause) Denote $c_t : \bigvee_{i=0}^{n-1} x_i = 1$, then c_t is called as a trivial clause over $\mathbb{Z}_2^n$ and the solution space of c_t is $\{(0, \cdots, 0)\}$.

From the perspective of the set, the degeneration case corresponds to the third case in Property 3.2. If a set can be characterized by a non-trivial clause, then there must exist x_i that does not appear in the clause, which is corresponded to the degenerated case. It means that the value of x_i does not influence whether the clause holds or not, hence when $s = (s_0, \cdots, s_i, \cdots, s_{n-1}) \in \overline{S}$, we will always have $s' = (s_0, \cdots, \overline{s}_i, \cdots, s_{n-1}) \in \overline{S}$. In this case, s and s' can be abstracted as a same point: $s^* := (s_0, \cdots, s_{i-1}, *, s_{i+1}, \cdots, s_{n-1}) \in \overline{S}$, which will be called mergeable later.

Based on the discussion above, it is clear that every clause corresponds to a trivial clause in a lower dimensional space. At the same time, only one point is cut by the trivial clause in the

lower dimensional space, and the corresponding points in the higher dimensional space must be a degenerated set that can be merged. Hence a sufficient and necessary condition of the plain set can be obtained by exploring the internal relationship between the set of mergeable points and the coefficients of variables in clauses.

3.2 A Sufficient and Necessary Condition of the Plain Set

From the previous discussions, we have established the connection between plain sets and degenerated sets. It can be concluded that S is plain only if $\overline{S}$ can degenerate to a lower dimensional set. Conversely, each plain set corresponds to either a single point or a degenerated set. To further elaborate on this relationship, let's start with the definition of mergeable.

Definition 3.2 (Mergeable) Suppose $S \in \mathbf{\Pi}_n$ and $i \in \mathbb{Z}_n$, if $s \in S$ and $s \oplus e_i \in S$, we say s and $s \oplus e_i$ are i-th position mergeable under S. Denote $s' = (s'_0, s'_1, \cdots, s'_{n-1})$, where

$$s'_j = \begin{cases} s_j, & \text{if } j \neq i; \\ *, & \text{if } j = i. \end{cases}$$

In this case, s' can represent the set $\{s, s \oplus e_i\}$.

After points merging, S can be regarded as a subset of $\mathbb{Z}_*^n := \{0, 1, *\}^n$, each $*$ represents two elements in $\mathbb{Z}_2^n$ whose specific components take over $\{0, 1\}$ and the rest have the same values. Further, other components can be merged simultaneously based on Definition 3.2. Suppose $x \in \mathbb{Z}_*^n$ and denote the number of $*$ in x as $wt_*(x)$. More generally, higher-order mergeable can be defined as below:

Definition 3.3 (k-order mergeable) Suppose $s^0, s^1, \cdots, s^{2^k-1} \in S$, if there exists $I \subseteq \mathbb{Z}_n$ such that these 2^k points in S can be represented as $s' = (s'_0, s'_1, \cdots, s'_{n-1})$, where

$$s'_i = \begin{cases} = s_i^0 = s_i^1 = \cdots = s_i^{2^k-1}, & \forall i \notin I; \\ *, & \forall i \in I, \end{cases}$$

and $wt_*(s') = k$, then $\{s^0, s^1, \cdots, s^{2^k-1}\}$ is called to be I-th position k-order mergeable under S, or for short, (I, k)-mergeable and I is called the mergeable position. At the same time, we still use s' to denote the set $\{s^0, s^1, \cdots, s^{2^k-1}\}$, then S can be regarded as a subset of $\mathbb{Z}_*^n$.

According to the above definition, a complementary set of a mergeable set can be easily characterized by a clause as below:

Property 3.3 Suppose s is a (I, k)-mergeable subset under S, then

$$c : \bigvee_{i \in \mathbb{Z}_n \setminus I} (x_i \oplus s_i) = 1 \tag{1}$$

is a FCC of $\overline{s}$.

Proof Substitute the value of s_i into Equation (1), then every expression in this clause is 0 and the clause is unsatisfiable; in reverse, if $x \in \mathbb{Z}_2^n$ makes c unsatisfiable, it must be the

case that every expression in the clause is 0, hence we have $x_i = s_i$, $\forall i \in Z_2^n \setminus I$, i.e., $x \in s$. In conclusion, we know that $\bar{c} = s$, hence c is a FCC of $\overline{s}$. □

Recall the discussion of degeneration in the previous section, a sufficient and necessary condition of the plain set can be obtained based on the properties of mergable sets.

Theorem 3.1 *(A Sufficient and Necessary Condition of the Plain Set)* $S \in \mathbf{\Pi}_n$ *is plain if and only if*

$$\overline{S} = \{s = (s_0, \cdots, s_{n-1}),\ \textit{where } s_i \in \{*, c_i\} \textit{ and } c = (c_0, \cdots, c_{n-1}) \textit{ is a constant depended on } \overline{S}\},$$

i.e., $\overline{S}$ *is a merageable set.*

The proof can be completed directly according to Property 3.3 and Definition 3.3. Theorem 3.1 provides a sufficient and necessary condition of the plain set and establishes the one-to-one correspondence between plain sets and mergeable sets. Moreover, a plain set can be constructed according to Property 3.3.

3.3 Plain Closure of Sets

For a non-plain set S, there will be more than one clause in its FCC, and S is the intersection of solution spaces of these clauses. In this case, each solution space of these clauses is a plain set containing S, which is called the plain closure of S.

Definition 3.4 (Plain Closure) Let $S \in \mathbf{\Pi}_n$ and $S' \in \mathbf{P}_n$, S' is called a plain closure of S if $S \subseteq S'$.

It is observed that $\mathbb{Z}_2^n$ is always a plain closure of all S's, thus the plain closure always exists, and we call $\mathbb{Z}_2^n$ to be trivial. If S is not plain, it can be expanded into plain sets by adding elements in $\overline{S}$. By constructing different closures of S and making their intersection S, a FCC of S can be obtained. Denote the set of all the plain closures of S as $\mathcal{C}(S)$. Furthermore, according to Theorem 3.1, all plain closures of a given set S can be obtained by exhausting all possible mergeable sets contained in $\overline{S}$ and collecting their complements. Each clause in a best FCC of S must correspond to a plain closure of S, so a best FCC of S is always a subset of $\mathcal{C}(S)$. Since our goal is to find the best FCC, we are particularly interested in the minimal plain closures of S, which are defined below.

Definition 3.2 (Minimal Plain Closure) Let $S \in \mathbf{\Pi}_n$ and S' be a plain closure of S. S' is minimal if $\nexists S'' \in \mathbf{P}_n$ such that $S \subseteq S'' \subset S'$.

Those non-minimal plain closures are said to be redundant which only increase the scale of the problem without adding any new information. Hence, it is crucial to eliminate redundancies to improve the efficiency of the SCP step.

Property 3.4 Suppose $c_1 : \bigvee_{i \in I_1} x_i \vee \bigvee_{j \in J_1} \overline{x}_j = 1$ and $c_2 : \bigvee_{i \in I_2} x_i \vee \bigvee_{j \in J_2} \overline{x}_j = 1$. If $I_1 \subset I_2$, $J_1 \subset J_2$, then $c_1 \subset c_2$.

Proof Suppose $x \in c_1$, then there exists either $i \in I_1 \subset I_2$ such that $x_i = 1$, or $j \in J_1 \subset J_2$ such that $x_j = 0$, hence we have $x \in c_2$. Then $c_1 \subset c_2$, which means that the former clauses will remove more infeasible points. □

Property 3.4 provides a sufficient condition for the minimal plain closures which can help remove the redundancy. After collecting all the minimal plain closures of S, we convert it into an SCP and call MILP solvers to get the minimum value, then a FCC can be obtained. In this process, we exhaust all the minimal plain closures of S without introducing new variables, so the FCC we constructed is a best FCC in theory. The details of the algorithm will be presented in the next section. Moreover, according to our comprehensive theoretical framework, it can be easily observed that the essential goal of both Boura's algorithm and QM algorithm is also to construct all the minimal plain closures, but different perspectives lead to different presentations of the algorithms. And our complete theoretical system can guarantee the optimality of FCCs.

3.4 The Relationship between FCC and FLIIC

In this subsection, we will further explore the relationship between FCCs and FLIICs, in order to reveal the difference and connection between MILP models and SAT models. First, a CNF clause can be naturally transformed into a linear integer inequality. Suppose $c : \bigvee_{i=0}^{n-1} (x_i \oplus c_i) = 1$, then

$$l : \sum_{i=0}^{n-1} (x_i + c_i - 2x_i c_i) \geq 1$$

has the same solution space with c.

Due to the wider range of coefficients of linear inequalities, not all inequalities can be converted into a CNF clause, that is, the plain set in the sense of the CNF clause is a subset of the plain set in the sense of the inequality. Next, we will further explore the relationship between them.

Firstly, the non-degenerate case is considered and the coefficients of the inequality are limited to $\{1, -1\}$, then the constant term has the following property:

Property 3.5 If the clause $c : \bigvee_{i\in I} x_i \vee \bigvee_{j\in J} \overline{x}_j = 1$ has the same solution space with the inequality $l : \sum_{i\in I} x_i - \sum_{j\in J} x_j + b \geq 0$, then $b = |J| - 1$.

Proof We prove it by contradiction. Suppose $s = (s_0, s_1, \cdots, s_{n-1}) \in \mathbb{Z}_2^n$.

- If $b < |J| - 1$, when all $s_j = 1$ and only one $s_i = 1$, we have $1 - |J| + b < 0$ and s is a feasible point of c. In this case, s is cut by l, which leads to a contradiction, hence $b \geq |J| - 1$;
- If $b \geq |J|$, when all $s_i = 0$ and all $s_j = 1$, we have $0 - |J| + b \geq 0$, hence s is a infeasible point of c. In this case, s is preserved by l, which leads to a contradiction, hence $b < |J|$;

In conclusion, we have $b = |J| - 1$. □

The aforementioned property implies that the constant term of the inequality is uniquely determined when the coefficients are the same. However, in practical applications, the coefficients of linear integer inequality are in a wider range, hence we will further explore the property of coefficients.

Property 3.6 Denote $c : \bigvee_{i\in I} x_i \vee \bigvee_{j\in J} \overline{x}_j = 1$ and

$$l : \sum_{i \in I} a_i x_i - \sum_{j \in J} a_j x_j + b \geq 0$$

have the same solution space in $\mathbb{Z}_2^n$, where $a_i, a_j \in \mathbb{Z}_n$, then

$$b + 1 \leq \sum_{j \in J} a_j \leq b + \min_{i \in I, j \in J} \{a_i, a_j\}.$$

Proof Suppose $s = (s_0, s_1, \cdots, s_{n-1})$. Next we consider some special cases to provide necessary conditions for coefficients:

- Let $s_i = 0$ for all $i \in I$ and $s_j = 1$ for all $j \in J$, then s is a infeasible point of c, hence $\sum_{j \in J} a_j > b$;
- Let $s_j = 1$ for all $j \in J$, for $i \in I$, suppose $s_i = 1$ and $s_{i'} = 0$, $\forall i' \in I \setminus \{i\}$, then s is a feasible point of c, hence $a_i - \sum_{j \in J} a_j + b \geq 0$, i.e., $\sum_{j \in J} a_j \leq b + \min_{i \in I} a_i$;
- Let $s_i = 0$ for all $i \in I$, for $j \in J$, suppose $s_j = 0$ and $s_{j'} = 1$, $\forall j' \in J \setminus \{j\}$, then s is a feasible point of c, hence $a_{j'} - \sum_{j \in J} a_j + b \geq 0$, i.e., $\sum_{j \in J} a_j \leq b + \min_{j \in J} a_j$.

In conclusion, we have:

$$b + 1 \leq \sum_{j \in J} a_j \leq b + \min_{i \in I, j \in J} \{a_i, a_j\}. \tag{2}$$

In addition, the above three conditions are compact boundary conditions, that is, the equality can be satisfied. □

The above discussion is carried out for the non-degenerate cases. When considering the degenerate cases, the clause and inequality are regarded as the full characterization of a degenerate set. Once a new variable x_n is introduced, the dimension of the space increases by 1, but in this case, the value of the newly added component has no affection on the value of the clause and the inequality. Hence the solution spaces remain unchanged and the conclusion still holds for degenerate cases.

By exploring the relationship between the coefficients of inequalities and the solution sets of clauses, we determine which integer linear inequalities can be transformed into clauses and establish a bidirectional relationship between the characterization of MILP and SAT. Such discussion will serve as a way to link MILP-based methods and SAT-based methods.

4 Algorithm to Get a FCC for a Given Set

In this section, based on the properties of plain sets discussed in the previous section, a new algorithm to construct a best FCC is proposed.

Given a set S, if it is plain, then a best FCC can be obtained directly according to Property 3.3. For a non-plain set S, we will construct plain closures by adding elements from $\overline{S}$ to S. From the point of view of $\overline{S}$, this operation is equivalent to removing infeasible points, i.e.,

taking subsets of $\overline{S}$. So the problem is reduced to finding all the maximal subsets of $\overline{S}$ whose complementary sets are plain. According to Theorem 3.1, this problem can be solved by finding all mergeable subsets of $\overline{S}$ which is the core of the algorithm. According to the definition, each $*$ corresponds to two elements that are only different in a certain component. Moreover, a k-order mergeable set contains 2^k elements, which can be obtained by merging two $(k-1)$-order mergeable sets. Therefore, a recursive method is adopted to construct high-order mergeable sets.

Denote $\overline{S}^0 := \overline{S}$, the first step is to collect all the 1-order mergeable sets in $\overline{S}$ by exhausting all components. Denote this procedure as ComputeMerge($\overline{S}^0, i$), $i = 0, \cdots, n-1$ and the result set as $\overline{S}^1 := \{s | s \in \overline{S}^0, s \oplus e_i \in \overline{S}^0, i \in \mathbb{Z}_n\}$. According to mergeable positions, $\overline{S}^1$ can be divided into the union of n sets:

$$\overline{S}^1 = \overline{S}^1_0 \cup \overline{S}^1_1 \cup \cdots \cup \overline{S}^1_{n-1},$$

where $\overline{S}^1_i = \{s | s \in \overline{S}^0, s \oplus e_i \in \overline{S}^0\}$. It should note that different $\overline{S}^1_i$ may have common elements which correspond to different mergeable positions. On step further, all the 2-order mergeable sets are the results of merging two elements in $\overline{S}^1_i$ at components $j \neq i$. Then by applying ComputeMerge$(\cdot, j)$ to each $\overline{S}^1_i$, all the 2-order mergeable sets can be obtained. More generally, the set $\overline{S}^k$ storing all the mergeable sets of the k-order can be recursively constructed as follows:

$$\overline{S}^k = \bigcup_{I \subseteq \mathbb{Z}_n, |I|=k-1, j \in \overline{I}} \overline{S}^k_{I \cup \{j\}} = \bigcup_{I \subseteq \mathbb{Z}_n, |I|=k-1, j \in \overline{I}} \{x | x \in \overline{S}^{k-1}_I, x \oplus e_j \in \overline{S}^{k-1}_I\},$$

where I represents the mergeable position of the $(k-1)$-order mergeable sets and $I \cup \{j\}$ represents the mergeable position of the newly constructed k-order mergeable sets. After constructing $S^0, S^1, \cdots, S^{n-1}$, all mergeable sets and their corresponding mergeable positions have been collected, and each corresponds to a plain closure of S. For details of the algorithm, see Algorithm 1. Then the algorithm to solve the SCP is called. Finally, according to Property 3.3, the FCCs of the selected plain closures can be obtained which form a best FCC for S.

Algorithm 1 Get all the minimal plain closures of S

Input: A set $S \in \mathbf{\Pi}_n$

Output: All the minimal plain closures of S

procedure ComputeMerge(A, i) **return** $\{x | x \in A, x \oplus e_i \in A\}$;

procedure ComputeAllClosure(S)

$\mathbb{R} := \varnothing$;

$S_0 := S$;

for $d = 1$ to $n - 1$ **do**

for $a \in \{x | wt(x) = d\}$ **do**

$k \xleftarrow{R} supp(a)$;

$S_a = ComputeMerge(S_{a \oplus e_k}, k)$;

for $a \in \{0, \cdots, n-1\}$ **do**

for $x \in S_a$ **do**

$R := \varnothing$;

for $k \in prec(a)$ **do**

$R = R \cup \{x \oplus k\}$;

$S_a = S_a \backslash \{x \oplus k\}$;

$\mathbb{R} = \mathbb{R} \cup \{R\}$;

return Minimize($\mathbb{R}$);

Remark 4.1 (The Time Complexity Analysis of Algorithm 1) The produce ComputeMerge() exhausts set A and search whether $x \oplus e_i$ is in A, hence the time complexity is $O(|A|)$ which is bounded by $O(2^n)$. Besides, while adopting a merge representations, this bound can be tightened to $O(2^{n-d})$, where d is the order of the mergeable set. In the produce ComputeAllClosure(), there are $\binom{n}{d}$ elements whose weight are d, hence the time complexity of the first loop is bounded by $O(\sum_{d=1}^{n} \binom{n}{d} \times 2^{n-d}) = O(3^n)$. In the second loop, the total number of iterations for the two outer loops is 2^n, corresponding to each inner loop having at most $2^{n-wt(x)}$ iterations, hence the time complexity is $O(\sum_{d=1}^{n} \binom{n}{d} \times 2^d) = O(3^n)$. In conclusion, the total time complexity is bounded by $O(3^n)$.

5 Applications

In this section, we apply the new algorithm to characterize S-boxes used in block ciphers. In practical applications, the target set can be divided into two types, those with probability and those without probability, corresponding to the DDT (LAT) and the *-DDT (*-LAT) respectively. To obtain the corresponding set, we introduce a Boolean function whose value is 1 when the transition is feasible and 0 otherwise. Please refer to subsection 2.2 for more details. In summary, the core of characterizations is constructing a FCC for a given set.

For characterizations of DDTs, extra variables will be appended to represent concrete probabilities. For example, to characterize the DDT of Present's S-box which has three non-zero probabilities: $P_0 = 1$, $P_1 = \frac{1}{2^2}$, $P_2 = \frac{1}{2^3}$, three binary variables are introduced, denoted as w_0,

w_1, w_2. Then tuples $(0,0,0)$, $(1,1,0)$, $(1,1,1)$ can represent all cases and the exact probability Pr can be calculated as $\log Pr = w_0 + w_1 + w_2$. In this case, the set of the DDT is a subsect of $\mathbb{Z}_2^{11}$.

There are two different modes for characterizations of *-DDTs when searching the minimal number of active S-boxes. For both of them, an extra binary variable A needs to be introduced for each S-box to indicate whether it is active. If the S-box is active, then $A = 1$; otherwise, $A = 0$. For a n-bit S-box, suppose $x_0, \cdots, x_{n-1}$ and $y_0, \cdots, y_{n-1}$ are variables which represent the input and output differences respectively. Denote the feasible point of the target *-DDT as S_0, then one can use $n+1$ inequalities to characterize the activeness of the S-box:

$$A \geq x_i, i = 0, \cdots, n-1;$$

$$x_0 + x_1 + \cdots + x_{n-1} \geq A.$$

In addition, a FCC of S_0 needs to be constructed. On the other hand, one can also construct a FCC of

$$S' = \{(x, y, A) | (x, y) \in S_0 \backslash \{(0, \cdots, 0)\}, A = 1\} \cup \{(0, \cdots, 0)\}$$

directly. The above two cases are denoted as *-DDT and A-DDT, and correspond to characterizations of subsets of $\mathbb{Z}_2^{2n}$ and $\mathbb{Z}_2^{2n+1}$ respectively.

In summary, there are three ways to characterize the differential transitions for an S-box, which correspond to three different sets. In experiments, we characterize these three sets for each block cipher and obtain their best FCCs using our new algorithm. Meanwhile, we also use Logic Friday to characterize the same sets for comparison. The experimental results are shown in Table 2. Compared with the existing method, our new algorithm can construct FCCs with fewer clauses, and the theoretical optimality can be proved for small-dimensional cases. For large S-boxes whose best FCCs can not be obtained, we can also get better solutions and provide a lower bound simultaneously. In addition, although the corresponding SCP is too large to get the optimal value for the *-DDTs of large S-boxes, we can still get better results based on the candidate set constructed by the new algorithm and algorithms to solve SCP. More specifically, we finally get a FCC with 7393 clauses for the *-DDT of AES S-box, which is the best result so far, compared to the previous result of 8302. For more results of the *-DDTs of large S-boxes, please refer to Table 3.

Table 2: Number of clauses to model differential transitions for various S-boxes

Cipher	LF *-DDT	Ours *-DDT	LF DDT	Ours DDT	LF A-DDT	Ours A-DDT
Rectangle	31	**30**	41	**40**	33	**33**
Present	39	**36**	55	**53**	43	**40**
Pyjamask	28	**28**	37	**36**	33	**32**
Piccolo	31	**31**	42	**39**	36	**36**

Panda	51	**49**	75	**69**	54	**53**
Noekeon	44	**44**	70	**64**	48	**47**
KNOT	32	**30**	42	**40**	35	**33**
Enocoro	44	**44**	70	**64**	48	**47**
Elephant	40	**38**	52	**50**	43	**41**
LblockS0-9	30	30	38/39	**37**	34	**34**
Skinny-64	31	31	42	**39**	36	**36**
Pride	31	31	41	**39**	36	**36**
Twine	47	**45**	71	**65**	51	**48**
Prince	52	**51**	73	**68**	55	**54**
Klein	45	**43**	65	**60**	49	**47**
Prost	31	31	41	**39**	36	**36**
Joltik	31	31	42	**39**	36	**36**
MIBS	52	**47**	78	**68**	53	**51**
Midori S0	47	47	57	**54**	48	**48**
Midori S1	57	**56**	78	**73**	58	**58**
Lillput	47	**45**	73	**65**	51	**48**
FBC	32	32	42	**40**	36	**36**
SC2000-4	45	**42**	70	**64**	46	**45**
ASCON-5	60	**59**	-	-	64	**64**
FIDES-5	125	**124**	-	-	127	**126**
Keccak	46	46	-	-	51	**51**
SC2000-5	125	**123**	-	-	126	**126**
Sycon	60	**59**	-	-	64	**64**
Shamash	125	125	-	-	130	**127**
DryGASCON-128	60	**59**	-	-	66	**64**
APN-6	297	**288**	-	-	298	**291**
FIDES-6	487	**455**	-	-	492	**460**
SC2000-6	615	**567**	-	-	615	**572**
KASUMI	2228	2000-**2026**	-	-	2233	2006-**2030**

[1] Columns with LF means the number of clauses in FCCs constructed by Logic Friday, columns with Ours means the number of clauses in FCCs constructed by Algorithm 1;

[2] Cells with '-' means that we did not do the corresponding experiments due to the encoding method of probabilities.

[3] For large-dimensional sets, we can not get the result of Logic Friday in a reasonable time in 'exact' mode. In this case, we adopt the outputs in 'fast' mode.

Table 3: Number of clauses to model *-DDTs for large S-boxes

Cipher	#Clause LF	#Clause Ours
AES	8302	7393
Camellia	8314	7428
ZUC S0	5307	4773
ZUC S1	8265	7399
Snow3G	7782	6800
MISTY-9	-	25373

Moreover, Sun *et al.* [34] accelerated the search of differential and linear characteristics with the SAT method and updated results for many lightweight block ciphers. For comparison, we choose two ciphers with open source codes *as comparisons and replace the FCCs of the S-boxes in their models with the best FCCs generated by our algorithm. To compare the runtime, anything else remains unchanged. The experimental results are shown in Table 4. It can be observed that best FCCs constructed by the new algorithm can accelerate the search process for some models. Furthermore, since all the minimal plain closures can be obtained, our work is able to completely control the number of clauses, allowing for the construction of different models to improve solving efficiency.

Table 4: Runtimes of automated search models

Cipher	Type	#Clause [34]	#Clause Ours	Time (s) [34]	Time (s) Ours
Present	DDT	43	**40**	30.60	37.57
Present	A-DDT	55	**53**	443.14	498.96
Present	LAT	51	**47**	230.88	**222.13**
Present	A-LAT	39	**38**	5.35	7.71
Lblock	DDT	38/39	**37**	629.99	**577.20**
Lblock	LAT	32	**30**	191.81	**190.21**

*Available at `https://github.com/SunLing134340/Accelerating_Automatic_Search.git`

References

[1] Eli Biham and Adi Shamir. Differential cryptanalysis of des-like cryptosystems. *Journal of CRYPTOLOGY*, 4(1):3–72, 1991. 1

[2] Mitsuru Matsui. Linear cryptanalysis method for des cipher. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 386–397. Springer, 1993. 1

[3] Mitsuru Matsui. On correlation between the order of s-boxes and the strength of des. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 366–375. Springer, 1994. 1

[4] LLC Gurobi Optimization. Gurobi optimizer reference manual. 2020. 1

[5] IBM ILOG Cplex. V12. 1: User's manual for cplex. *International Business Machines Corporation*, 46(53):157, 2009. 1

[6] Ksenia Bestuzheva, Mathieu Besançon, Wei-Kun Chen, Antonia Chmiela, Tim Donkiewicz, Jasper van Doornmalen, Leon Eifler, Oliver Gaul, Gerald Gamrath, Ambros Gleixner, Leona Gottwald, Christoph Graczyk, Katrin Halbig, Alexander Hoen, Christopher Hojny, Rolf van der Hulst, Thorsten Koch, Marco Lübbecke, Stephen J. Maher, Frederic Matter, Erik Mühmer, Benjamin Müller, Marc E. Pfetsch, Daniel Rehfeldt, Steffan Schlein, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Boro Sofranac, Mark Turner, Stefan Vigerske, Fabian Wegscheider, Philipp Wellner, Dieter Weninger, and Jakob Witzig. The SCIP Optimization Suite 8.0. Technical report, Optimization Online, December 2021. 1

[7] Shengbao Wu and Mingsheng Wang. Security evaluation against differential cryptanalysis for block cipher structures. *Cryptology ePrint Archive*, 2011. 1

[8] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and linear cryptanalysis using mixed-integer linear programming. In *International Conference on Information Security and Cryptology*, pages 57–76. Springer, 2011. 1

[9] 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 *International Conference on the Theory and Application of Cryptology and Information Security*, pages 158–178. Springer, 2014. 1, 1.1

[10] Kai Fu, Meiqin Wang, Yinghua Guo, Siwei Sun, and Lei Hu. Milp-based automatic search algorithms for differential and linear trails for speck. In *International Conference on Fast Software Encryption*, pages 268–288. Springer, 2016. 1

[11] Zejun Xiang, Wentao Zhang, Zhenzhen Bao, and Dongdai Lin. Applying milp method to searching integral distinguishers based on division property for 6 lightweight block ciphers. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 648–678. Springer, 2016. 1

[12] Cui Tingting, Chen Shiyao, Fu Kai, Wang Meiqin, and Jia Keting. New automatic search tool for impossible differentials and zero-correlation linear approximations. *SCIENCE CHINA Information Sciences*, 2016. 1

[13] Yu Sasaki and Yosuke Todo. New algorithm for modeling s-box in milp based differential and division trail search. In *International Conference for Information Technology and Communications*, pages 150–165. Springer, 2017. 1

[14] Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. A security analysis of deoxys

and its internal tweakable block ciphers. *IACR Transactions on Symmetric Cryptology*, pages 73–107, 2017. 1

[15] Yosuke Todo, Takanori Isobe, Yonglin Hao, and Willi Meier. Cube attacks on non-blackbox polynomials based on division property. *IEEE Transactions on Computers*, 67(12):1720–1736, 2018. 1

[16] Niklas Eén and Niklas Sörensson. An extensible sat-solver. In *International conference on theory and applications of satisfiability testing*, pages 502–518. Springer, 2003. 1

[17] Mate Soos, Karsten Nohl, and Claude Castelluccia. Extending sat solvers to cryptographic problems. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 244–257. Springer, 2009. 1

[18] SEPARATE DECISION QUEUE. Cadical at the sat race 2019. *SAT RACE 2019*, page 8, 2019. 1

[19] Nicky Mouha and Bart Preneel. Towards finding optimal differential characteristics for arx: Application to salsa20. *Cryptology ePrint Archive*, 2013. 1

[20] Stefan Kölbl, Gregor Leander, and Tyge Tiessen. Observations on the simon block cipher family. In *Annual Cryptology Conference*, pages 161–185. Springer, 2015. 1

[21] Yunwen Liu, Qingju Wang, and Vincent Rijmen. Automatic search of linear trails in arx with applications to speck and chaskey. In *International Conference on Applied Cryptography and Network Security*, pages 485–499. Springer, 2016. 1, 1.1

[22] Ling Sun, Wei Wang, and Meiqin Wang. More accurate differential properties of led64 and midori64. *IACR Transactions on Symmetric Cryptology*, pages 93–123, 2018. 1, 1.1

[23] Yu Liu, Huicong Liang, Muzhou Li, Luning Huang, Kai Hu, Chenhe Yang, and Meiqin Wang. Stp models of optimal differential and linear trail for s-box based ciphers. *Cryptology ePrint Archive*, 2019. 1

[24] Ling Sun, Wei Wang, and Meiqin Wang. Automatic search of bit-based division property for arx ciphers and word-based division property. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 128–157. Springer, 2017. 1

[25] Sage Developers. Sagemath, the sage mathematics software system (version 9.0). 2020. 1.1

[26] Christina Boura and Daniel Coggia. Efficient milp modelings for sboxes and linear layers of spn ciphers. *IACR Transactions on Symmetric Cryptology*, pages 327–361, 2020. 1.1

[27] Aleksei Udovenko. Milp modeling of boolean functions by minimum number of inequalities. Cryptology ePrint Archive, Report 2021/1099, 2021. 1.1

[28] Ting Li and Yao Sun. Superball: A new approach for milp modelings of boolean functions. *IACR Transactions on Symmetric Cryptology*, pages 341–367, 2022. 1.1

[29] Xiutao Feng, Yu Tian, Yongxing Wang, Shengyuan Xu, and Anpeng Zhang. Full linear integer inequality characterization of sets over $\mathbb{Z}_2^n$. `http://www.chinaxiv.org/abs/202210.00055v2`. 1.1

[30] Ahmed Abdelkhalek, Yu Sasaki, Yosuke Todo, Mohamed Tolba, and Amr M Youssef. Milp modeling for (large) s-boxes to optimize probability of differential characteristics. *IACR Transactions on Symmetric Cryptology*, pages 99–129, 2017. 1.1

[31] 'logic friday', http://sontrak.com/. 1.1

[32] Willard V Quine. The problem of simplifying truth functions. *The American mathematical monthly*, 59(8):521–531, 1952. 1.1

[33] Robert K Brayton, Gary D Hachtel, Curt McMullen, and Alberto Sangiovanni-Vincentelli. *Logic minimization algorithms for VLSI synthesis*, volume 2. Springer Science & Business Media, 1984.

1.1

[34] Ling Sun, Wei Wang, and Meiqin Wang. Accelerating the search of differential and linear characteristics with the sat method. *IACR Transactions on Symmetric Cryptology*, pages 269–315, 2021. 1.1, 5, 4

[35] Senpeng Wang, Dengguo Feng, Bin Hu, Jie Guan, Tairong Shi, and Kai Zhang. The simplest sat model of combining matsui's bounding conditions with sequential encoding method. *Cryptology ePrint Archive*, 2022. 1.1

[36] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2022. 2.4

[37] Zhouxing Su, Qingyun Zhang, Zhipeng Lü, Chu-Min Li, Weibo Lin, and Fuda Ma. Weighting-based variable neighborhood search for optimal camera placement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12400–12408, 2021. 2.4

[38] Weibo Lin, Fuda Ma, Zhouxing Su, Qingyun Zhang, Chumin Li, and Zhipeng Lü. Weighting-based parallel local search for optimal camera placement and unicost set covering. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, pages 3–4, 2020. 2.4