IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2022, No. 3, pp. 341–367. DOI:10.46586/tosc.v2022.i3.341-367

SuperBall: A New Approach for MILP Modelings of Boolean Functions

Ting Li[1] and Yao Sun[1,2] (✉)

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{liting,sunyao}@iie.ac.cn

Abstract. Mixed Integer Linear Programming (MILP) solver has become one of the most powerful tools of searching for cryptographic characteristics. It has great significance to study the influencing factors of the efficiency of MILP models. For this goal, different types of MILP models should be constructed and carefully studied. As Boolean functions are the fundamental cryptographic components, in this paper, we study the descriptive models of Boolean functions. Here, a descriptive model of a Boolean function refers to a set of integer linear inequalities, where the set of the binary solutions to these inequalities is exactly the support of this Boolean function. Previously, it is hard to construct various types of descriptive models for study, one important reason is that only a few kinds of inequalities can be generated. On seeing this, a new approach, called SuperBall, is proposed to generate inequalities. The SuperBall approach is based on the method of undetermined coefficients, and it could generate almost all kinds of inequalities by appending appropriate constraints. Besides, the Sasaki-Todo Algorithm is also improved to construct the descriptive models from a set of candidate inequalities by considering both their sizes and strengths, while the strengths of descriptive models have not been considered in the previous works. As applications, we constructed several types of descriptive models for the Sboxes of Liliput, SKINNY-128, and AES. The experimental results first prove that the diversity of the inequalities generated by the SuperBall approach is good. More importantly, the results show that the strengths of descriptive model do affect the efficiencies, and although there is not a type of descriptive model having the best efficiency in all experiments, we did find a specific type of descriptive model which has the minimal size and relatively large strength, and the descriptive models of this type have better efficiencies in most of our experiments.

Keywords: MILP · Inequality · Boolean function · Sbox · linear layer

1 Introduction

In 2009, Borghoff et al. firstly applied the Mixed Integer Linear Programming (MILP) method to attack Bivium [BKS09]. Although they did not get outstanding results at that time, they believed that this kind of attacks was also applicable to other cryptographic algorithms. In 2011, the MILP method was used for cryptanalyses by Mouha et al. [MWGP11] and Wu et al. [WW11] again. The authors transformed the problem of counting the minimal number of active S-boxes into an MILP problem. Mouha et al. proposed the MILP modelings for the XOR operation and the linear transformations, but they could not model the differential properties of Sboxes. Thus, only word-oriented models were constructed, and these models cannot be applied to all kinds of ciphers at that time, e.g. Present [BKL+07] and Gift [BPP+17]. In 2014, Sun et al. proposed

Received: 2022-03-01 Revised: 2022-06-01 Accepted: 2022-08-01 Published: 2022-09-09

two methods to model the possible and impossible differential propagations of the Sboxes in [SHW+14a, SHW+14b], so they could find the differential characteristics automatically for bit-oriented block ciphers. Since then, MILP methods have been widely applied in cryptanalysis, including differential attacks [SHW+14a, SHW+14b, FWG+16, ZZDX20], impossible differential attacks [ST17b], and cube attacks [XZBL16, SWW17, TIHM18, HW18, WHG+19, HLM+20].

Usually, the propagation property of a cryptographic component (e.g. Sbox) can be represented as a Boolean function. For a Boolean function F, to model F by an MILP method is to compute a set of integer linear inequalities such that the set of binary solutions to these inequalities is exactly the support of F. For convenience, we call the set of inequalities that models F as **a descriptive model** of F. According to [BC20], two problems should be solved if we want to compute a descriptive model of F.

Problem 1: How to generate inequalities such that the bit vectors in the support of F are all solutions to these inequalities?

Problem 2: If the first problem is solved, then among the candidate inequalities, how to choose a set of inequalities that models F?

For Problem 1, Sun et al. proposed two methods to generate inequalities in [SHW+14a, SHW+14b], the convex hull method and the logical condition method. In the convex hull method, the inequalities were generated by computing the H-representation of the convex hull of all points corresponding to possible differential propagations. The convex hull can be computed by the Sage software [Dev16]. The logical condition method could generate an inequality for every point corresponding to the *impossible* differential propagation. However, both of the above methods can only work on small Sboxes, e.g. 4-bit Sboxes. In 2017, Abdelkhalek et al. gave the first MILP model for 8-bit Sboxes in [AST+17], they studied the product-of-sum representation and generated the inequalities via the Espresso algorithm. In 2020, Boura and Coggia proposed several techniques to generate inequalities for Sboxes and linear layers in [BC20]. They extended the logical condition method to exclude more points, and considered the inequalities with zero coefficients. Here, the points excluded by an inequality refer to the points that are not solutions of this inequality. They also constructed a kind of new inequalities, called distorted balls. More importantly, the authors proposed a method of generating new inequalities from existing inequalities. And their experimental results also showed that Boura and Coggia's methods generated more kinds of inequalities than the previous works. Although Boura and Coggia's methods could still only generate some special kinds of inequalities, their work greatly motivated the following studies. In 2021, Udovenko proposed a new approach to generate inequalities in [Udo21]. His idea is similar to ours, but the details are distinct. Both Udovenko's work and ours were done independently almost at the same time. We will provide more details about Udovenko's work in the end of this section.

For Problem 2, Sun et al. gave a greedy method to remove redundant inequalities in [SHW+14b]. Their idea is to iteratively select the inequality, which excludes the most points, from the set of unselected inequalities. Sun et al.'s method could reduce the size of a descriptive model, but cannot reach the optimal size. In 2017, Sasaki and Todo proposed a new algorithm to reduce the size of a descriptive model automatically in [ST17a]. Specifically, in order to choose a set of inequalities that models a Boolean function F, a binary variable is assigned to each candidate inequality to show if this inequality is selected, and a constraint is added to each bit vector that is outside the support of F. These constraints ensure that each bit vector outside the support of F is rejected by at least one inequality. Then, an MILP model is constructed to find out a descriptive model with the specified size. Generally, the objective of the MILP model is to minimize the number of inequalities, so the inequalities which reject as many bit vectors outside the support of F as possible are preferred. The Sasaki-Todo algorithm has been widely used in the follow-up research.

Motivations: Since MILP models have been widely used in cryptanalyses, it is important to improve the efficiency of solving MILP models. However, it still remains unclear which type of descriptive model could achieve the best efficiency. Previously, the descriptive models that have minimal sizes were assumed to have better efficiencies, but some counter-examples in [ST17a] showed that this type of descriptive model may not be so efficient sometimes. To resolve this problem, various types of descriptive models should be constructed and carefully studied. And for this goal, the first step is to generate various inequalities, because all descriptive models are constructed from a set of candidate inequalities. If the candidate inequalities have less diversity, the differences in the efficiencies between the descriptive models will not be so obvious, which may prevent us from finding out the most efficient descriptive models.

Contributions: Firstly, we propose a novel approach, called the SuperBall[1] approach, to generate inequalities for Boolean functions and the sets of inequalities generated by the SuperBall approach have greater diversities than those in previous. Existing methods could only generate some special kinds of inequalities. Unlike the previous works, the SuperBall approach generates inequalities in a new way. By using the *method of undetermined coefficients*, we regard the coefficients and constant terms of the inequalities as unknowns, and formulate the constraints that the unknowns should satisfy. Then new inequalities can be obtained by solving the constructed systems. To make the constructed systems easier to study as well as to speed up the solving procedure, we use the *divide-and-conquer* strategy. Specifically, we divide the set of general inequalities into several patterns, and in each pattern, the signs of coefficients in the inequalities are fixed, which makes the related studies much easier. The detailed studies can be found in Sec. 3. In this paper, we use the concept of diversity to evaluate a set of inequalities. If the number of inequalities is large and the inequalities generated are “different” from each other, we say the set of inequalities has a great diversity. To measure the diversity of the sets of inequalities, we define the diversity index in Definition 5. We compared the diversity indexes of the sets of inequalities generated by our approach with those of [SHW$^+$14b] and [AST$^+$17] in Table 1. Because the candidate inequalities of [BC20] and [Udo21] were not available, we generated some other descriptive models with minimal sizes instead. The comparisons are shown in Table 2. The results show that the set of inequalities generated by the SuperBall approach has a greater diversity than previous works.

Secondly, we improve the Sasaki-Todo algorithm by considering both the sizes and strengths of descriptive models, while the strengths of models have not been considered before. “Strength”, also called “tightness” in some papers (e.g. [MELR13]), is defined as the search space (relaxed feasible region) that the solver needs to explore in order to find the (optimal integer) solution. The descriptive model of a Boolean function that has the largest strength is just the convex hull of the support of the Boolean function. However, the size, i.e. the number of inequalities, of the convex hull is large. In [Vie15], Vielma pointed out that an efficient descriptive model should balance the size and the strength. But since the strength is hard to be controlled when we are selecting inequalities from a candidate set, we define the concepts of approximate strength and cover rate for a descriptive model. Then we improve the Sasaki-Todo algorithm by bounding the approximate strength and maximizing the cover rate at the same time. Specifically, given an approximate strength $s \geq 0$, the improved Sasaki-Todo algorithm computes a descriptive model that has the minimal size and a relatively large cover rate among the ones whose approximate strengths are at least s. The definitions of approximate strength and cover rate, as well as the details about the improved Sasaki-Todo algorithm come in Sec. 4.

Thirdly, we find and suggest a type of descriptive model which has a better efficiency

[1]In [BC20], Boura and Coggia’s methods of generating inequalities were called standard or distorted ball methods. As our approach could generate more kinds of inequalities, following the names of standard or distorted balls, we call our approach as the SuperBall approach.

in most of our experiments for the first time, although there is not a type of descriptive models having the best efficiency in all experiments. The great diversity of the set of inequalities enables us to construct several different types of descriptive models for the Sboxes of Liliput, SKINNY-128, and AES, where the types of descriptive models are mainly defined by their strengths. We designed two search tasks, verifying the differential pairs and finding the minimal numbers of active Sboxes, to test the efficiencies of descriptive models. As a result, we found the strengths of descriptive models do affect the efficiencies, and the descriptive models that have minimal sizes and relatively high cover rates often have better efficiencies in our experiments.

Our conclusion seems to contradict with the previous counter-examples in which showed that the descriptive models minimal sizes may not be so efficient sometimes, e.g. experiments in [ST17a]. We think we can explain this phenomenon in two aspects. Firstly, the sets of inequalities generated before the work [BC20] lack good diversity, and the strength factor related to inequalities is also not considered when the inequalities are generated, so we think the diversity of the candidate inequalities used previously is not great enough. Secondly, there exist many descriptive models that have the same minimal size, but the strengths of these descriptive models are distinct. Among these minimal descriptive models, the ones with the cover rate 0 may lead to much worse efficiencies than the ones with high cover rates in our experiments. So we guess the descriptive models used in the counter-examples perhaps have low cover rates.

Relation to Udovenko's work in [Udo21]: Udovenko proposed a similar idea for generating inequalities concurrently and independently of our work. The original manuscript of our paper [Sun21] was submitted to eprint slightly earlier than Udovenko's paper. Besides the high-level ideas, Udovenko also used the same divide-and-conquer strategy to study the inequalities in specific patterns, but his method of calculating the inequalities is different from our work. Specifically, in order to compute all maximal sets of points that can be removed by a single inequality (referred as maximal 1-removable sets in [Udo21]), Udovenko utilized the techniques for learning monotone Boolean functions. In the enumeration procedure, he first constructed an oracle to check whether a set is 1-removable, and then the maximal 1-removable sets can be generated by calling the oracle and updating some 1-removable sets with special properties for a few times, the constrains for these sets can be generated in the same time. But in this paper, we use the method of undetermined coefficients such that the above inequalities can be computed by only one call of the MILP solver. Besides the advantage in the efficiency, the method of undetermined coefficients also gives us a great flexibility to generate various kinds inequalities, e.g. the inequalities that balance the size and strength factors, as well as the inequalities containing many zero coefficients.

All source codes of the algorithms in this paper, the inequalities in the descriptive models, and the MILP models used in the experiments, can be found at `https://github.com/ysun0102/superball`.

Organization: Some necessary notations and preliminaries are presented in Sec. 2. We introduce the SupberBall approach for generating inequalities in Sec. 3, and the improved Sasaki-Todo algorithm is presented in Sec. 4. Experiments are shown in Sec. 5. We conclude this paper in Sec. 6.

2 Preliminaries

2.1 Notations

Let F be a Boolean function from $\mathbb{F}_2^n$ to $\mathbb{F}_2$, where $\mathbb{F}_2$ is the field consisting of $\{0,1\}$. In this paper, the n-bit vector $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1}) \in \mathbb{F}_2^n$ is also called as a **point** and written as $v_0 v_1 \cdots v_{n-1}$ for short. The **support** of F consists of the points on which F

takes 1, i.e. $\mathrm{Supp}(F) := \{\mathbf{v} \in \mathbb{F}_2^n \mid F(\mathbf{v}) = 1\}$. Let $\mathrm{wt}(\mathbf{v})$ denote the Hamming weight of the point $\mathbf{v}$.

The inequality discussed in this paper always has the following linear form

$$a_0x_0 + a_1x_1 + \cdots + a_{n-1}x_{n-1} + b \geq 0, \tag{1}$$

where a_i's and b are integers in $\mathbb{Z}$ and $x_i \in \{0, 1\} \subset \mathbb{Z}$ for $0 \leq i < n$. Please note that, in the Boolean function F, operations are *bit AND* and *XOR* ($\oplus$), while in the inequality (1), *integer multiplication* and *addition* (+) are used. For sake of simplification, we also say $x_i \in \mathbb{F}_2$ if no confusion arises.

Denote the linear polynomial $a_0x_0 + a_1x_1 + \cdots + a_{n-1}x_{n-1} + b$ by $f \in \mathbb{Z}[x_0, \ldots, x_{n-1}]$. Studying the inequality (1) actually equals to studying the polynomial f. The value of f taking on $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1}) \in \mathbb{F}_2^n$ is the integer $a_0v_0 + \cdots + a_{n-1}v_{n-1} + a_n \in \mathbb{Z}$. The set of **solutions** to an inequality $f \geq 0$ is defined as $\mathrm{Sol}(f \geq 0) := \{\mathbf{x} \in \mathbb{F}_2^n \mid f(\mathbf{x}) \geq 0\}$, and the set of common solutions to several inequalities, say $\{f_0 \geq 0, \ldots, f_m \geq 0\}$, is $\mathrm{Sol}(f_0 \geq 0, \ldots, f_m \geq 0) := \{\mathbf{x} \in \mathbb{F}_2^n \mid f_i(\mathbf{x}) \geq 0, \text{ for } 0 \leq i \leq m\}$.

Therefore, to **model a Boolean function** F with inequalities, is to find a set of inequalities $\{f_0 \geq 0, \ldots, f_m \geq 0\}$ such that $\mathrm{Sol}(f_0 \geq 0, \ldots, f_m \geq 0) = \mathrm{Supp}(F)$. And we call the set $\{f_0 \geq 0, \ldots, f_m \geq 0\}$ that models F as a **descriptive model** of F.

The *strength* (also called tightness) of a descriptive model reflects the scale of the search space (relaxed feasible region) that the solver needs to explore in order to find the optimal integer solution [MELR13]. The smaller the search space is, the greater the strength of a descriptive model has. To measure the strength factor related to an inequality, we define the strength index of an inequality $f \geq 0$ as the cardinality of the set $\mathrm{Sol}(f = 0)$. Figure 1 shows that if the sets $\mathrm{Sol}(f = 0)$ of inequalities are larger, the descriptive model constructed by these inequalities may have a larger strength. Given two sets of inequalities $P = \{f_i \geq 0, 0 \leq i \leq 2\}$ and $P' = \{f_i' \geq 0, 0 \leq i \leq 2\}$, for any inequality $f_i \geq 0$ in P, there always exists an inequality $f_i' \geq 0$ in P' such that $\mathrm{Sol}(f_i < 0) = \mathrm{Sol}(f_i' < 0)$ and $\mathrm{Sol}(f_i = 0) \subset \mathrm{Sol}(f_i' = 0)$. The descriptive models constructed by P and P' have the same integer solutions. However, the search space related to P is larger than that of P' as shown in Figure 1, so solving the model constructed by P ofter costs more time.

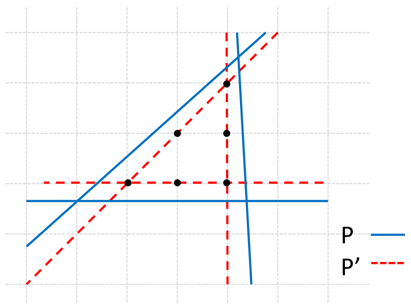

Figure 1: The search spaces of the models constructed by P and P'.

To construct good descriptive models which balance the size and the strength, for an inequality $f \geq 0$, we also consider the set $\mathrm{Sol}(f < 0) := \{\mathbf{x} \in \mathbb{F}_2^n \mid f(\mathbf{x}) < 0\}$ and the set $\mathrm{Sol}(f = 0) := \{\mathbf{x} \in \mathbb{F}_2^n \mid f(\mathbf{x}) = 0\}$. We will see in Sec. 4 that an inequality $f \geq 0$ with a large $\mathrm{Sol}(f < 0)$ is usually selected into the minimal descriptive model.

2.2 Some Boolean functions to be modeled

In automatic search models, propagation patterns of components need to be represented by MILP models. We consider three kinds of Boolean functions that are deduced from the

Differential Distribution Tables (DDTs) of Sboxes, linear layers, and division property.

Boolean functions from DDTs of Sboxes

Generally, DDTs are used to describe the differential propagation of Sboxes. For an m-bit Sbox S, let $\alpha, \beta \in \mathbb{F}_2^m$ be input and output differential pair. Then $\mathrm{DDT}(\alpha, \beta)$ is the cardinality of the set $\{\mathbf{x} \in \mathbb{F}_2^m \mid S(\mathbf{x}) \oplus S(\mathbf{x} \oplus \alpha) = \beta\}$. The Boolean function deduced from the DDT of S is:

$$(\alpha, \beta) \in \mathbb{F}_2^{2m} \mapsto \begin{cases} 1, \text{if } \mathrm{DDT}(\alpha, \beta) > 0, \\ 0, \text{if } \mathrm{DDT}(\alpha, \beta) = 0. \end{cases}$$

Boolean functions from linear layers

For a linear layer, the output $\mathbf{y} \in \mathbb{F}_2^m$ can usually be represented as a linear transform of the input $\mathbf{x} \in \mathbb{F}_2^m$, i.e. $\mathbf{y}^T = M\mathbf{x}^T$, where M is an $m \times m$ matrix in $\mathbb{F}_2^{m \times m}$. Thus, the Boolean function deduced from the linear layer is:

$$(\mathbf{x}, \mathbf{y}) \in \mathbb{F}_2^{2m} \mapsto \begin{cases} 1, \text{if } M\mathbf{x}^T = \mathbf{y}^T, \\ 0, \text{if } M\mathbf{x}^T \neq \mathbf{y}^T. \end{cases}$$

Boolean functions from division property

As the bit-based division property can propagate like the differential characteristics [XZBL16], the Boolean functions can be deduced similarly by enumerating all feasible input and output pairs. However, since the algebraic structures of the propagations are clear, the division property is often modeled by inequalities that use additional auxiliary variables, e.g. [TIHM18, HW18, WHG+19, HLM+20, HSWW20].

3 The SuperBall approach for generating inequalities

Let F be a Boolean function from $\mathbb{F}_2^n$ to $\mathbb{F}_2$, and S be the set $\mathrm{Supp}(F) \subset \mathbb{F}_2^n$. To model the boolean function F, we need to find a set of inequalities such that the set of their common solutions is exactly S. For this aim, we first propose a new approach, called SuperBall, to generate various inequalities in this section. Next, we improve Sasaki-Todo algorithm for selecting inequalities in the next section.

To illustrate our approach, we use the following toy example throughout this paper.

Example 1. Let $S = \{1000, 1010, 0110, 1110, 1001, 0101, 1101\} \subset \mathbb{F}_2^4$ be the support of a Boolean function F, where 1101 is short for the point $(1, 1, 0, 1) \in \mathbb{F}_2^4$. We want to compute a set of inequalities that models F, or equivalently, to compute a set of linear polynomials $\{f_0, f_1, \ldots, f_m\} \subset \mathbb{Z}[x_0, x_1, x_2, x_3]$, such that $\mathrm{Sol}(f_0 \geq 0, \ldots, f_m \geq 0) = S$.

3.1 Main ideas

Given a set $S \in \mathbb{F}_2^n$, to compute an inequality such that the points in S are all solutions to this inequality, our first idea is to use the *method of undetermined coefficients*. Specifically, in Example 1, to solve for a polynomial $f = a_0x_0 + \cdots + a_3x_3 + b$ such that $S \subset \mathrm{Sol}(f \geq 0)$, we regard a_0, a_1, a_2, a_3, b as unknowns in $\mathbb{Z}$ and solve the following system:

$$\begin{cases} f(1000) = & a_0 + b & \geq 0, \\ f(1010) = & a_0 + a_2 + b & \geq 0, \\ f(0110) = & a_1 + a_2 + b & \geq 0, \\ f(1110) = & a_0 + a_1 + a_2 + b & \geq 0, \\ f(1001) = & a_0 + a_3 + b & \geq 0, \\ f(0101) = & a_1 + a_3 + b & \geq 0, \\ f(1101) = & a_0 + a_1 + a_3 + b & \geq 0. \end{cases} \tag{2}$$

The system (2) consists of $|S|$ constraints. Clearly, this system has a huge number of solutions in $\mathbb{Z}^5$, and each solution $(\tilde{a}_0, \tilde{a}_1, \tilde{a}_2, \tilde{a}_3, \tilde{b}) \in \mathbb{Z}^5$ deduces a polynomial $\tilde{f} = \tilde{a}_0 x_0 + \cdots + \tilde{a}_3 x_3 + \tilde{b}$, as well as an inequality $\tilde{f} \geq 0$. According to the constraints in (2), we must have $S \subset \mathrm{Sol}(\tilde{f} \geq 0)$. However, most solutions of the system (2) are useless. For example, if there are two deduced polynomials, say f and f', such that $\mathrm{Sol}(f \geq 0) = \mathrm{Sol}(f' \geq 0)$ and $\mathrm{Sol}(f = 0) = \mathrm{Sol}(f' = 0)$, then one of them is useless.

Moreover, in practical applications, we usually have extra requests on the inequalities. For example, we often expect a polynomial f such that the set $\mathrm{Sol}(f < 0)$ is as large as possible, since this kind of inequalities is useful to reduce the sizes of descriptive models. To satisfy this request, we hope the following constraints to hold as many as possible:

$$\begin{array}{llll} f(0000) = & b < 0, & f(0100) = a_1 + b < 0, & \\ f(1100) = & a_0 + a_1 + b < 0, & f(0010) = a_2 + b < 0, & \\ f(0001) = & a_3 + b < 0, & f(0011) = a_2 + a_3 + b < 0, & (3) \\ f(1011) = & a_0 + a_2 + a_3 + b < 0, & f(0111) = a_1 + a_2 + a_3 + b < 0, & \\ f(1111) = & a_0 + a_1 + a_2 + a_3 + b < 0. & & \end{array}$$

The number of constraints in (3) is $2^n - |S|$.

Please note that, unlike the constraints in (2) that should hold at the same time, only some of the constraints in (3) can hold generally. And for the latter case, the difficulty is how to determine which of these constraints could hold simultaneously. In [Udo21], the author used a testing method to resolve this problem. That is, some constraints in (3) are chosen first, and then combined with the constraints in (2), a test is performed to check whether there is a solution to the constructed system. As the number of constraints in (3) is often large, this method is not very efficient. We use a simpler method to resolve this problem in Sec. 3.3.

The method of undetermined coefficients is able to compute a polynomial f such that $S \subset \mathrm{Sol}(f \geq 0)$, but it may not be efficient because the numbers of constraints in (2) and (3) are often large, which makes the constructed system hard to solve. So our second idea is to use the *divide-and-conquer strategy.* Note that the difference between the inequalities $a_0 + b \geq 0$ and $a_0 + a_2 + b \geq 0$ in the system (2) is only a_2. If we assume $a_2 \geq 0$, then the former inequality implies the latter one. This means the numbers of constraints in (2) and (3) can be reduced significantly if we know the signs of the coefficients a_i's.

Thus, instead of computing a polynomial $f = a_0 x_0 + \cdots + a_3 x_3 + b$ when the coefficients a_j's are in $\mathbb{Z}$, we only consider a specific *pattern* of f each time by fixing the signs of a_j's.

Definition 1. For a polynomial $f = a_0 x_0 + \cdots + a_{n-1} x_{n-1} + b \in \mathbb{Z}[x_0, \ldots, x_{n-1}]$, we say f is in a **pattern** if there exists a set $P \subset \{0, 1, \ldots, n-1\}$ such that $a_j \geq 0$ for $j \in P$ and $a_j \leq 0$ for $j \notin P$. Let $\mathbf{c} = (c_0, c_1, \ldots, c_{n-1})$ be a point in $\mathbb{F}_2^n$ such that $c_j = 0$ for $j \in P$ and $c_j = 1$ for $j \notin P$. We call $\mathbf{c}$ is the **center** of this pattern. For simplicity, we also say f is in the pattern $\mathbf{c}$.

If a polynomial f is in the pattern $\mathbf{c}$, then f takes the minimal value on the center $\mathbf{c}$, since $x_i \in \{0, 1\}$ for $0 \leq i < n$. For example, one pattern of $f = a_0 x_0 + \cdots + a_3 x_3 + b$ is $\{a_0 \geq 0,\ a_1 \geq 0,\ a_2 \leq 0,\ a_3 \leq 0\}$ and the center of this pattern is 0011. Then f takes the minimal value on the center point 0011, i.e. $x_0 = 0, x_1 = 0, x_2 = 1, x_3 = 1$. Particularly, a polynomial may be in several patterns if some of its coefficients are zeros.

By definition, there are 2^n patterns in all. To solve for all possible polynomials (as well as inequalities), we can perform the computations in each pattern first, and then gather all results. In fact, this divide-and-conquer strategy not only speeds up the computations, but also makes the study of inequalities easier in each pattern. Because by the following proposition and corollaries, to compute a polynomial in some pattern $\mathbf{c}$, it suffices to compute a polynomial in the pattern $\mathbf{0}$.

Proposition 1. *Let $\bar{f} = \bar{a}_0x_0 + \cdots + \bar{a}_{n-1}x_{n-1} + \bar{b}$ be a polynomial in the pattern $\mathbf{0}$. Let $\mathbf{c} = (c_0, \ldots, c_{n-1})$ be a point in $\mathbb{F}_2^n$, and P be the subset of $\{0, 1, \ldots, n-1\}$ such that $c_j = 0$ for $j \in P$ and $c_j = 1$ for $j \notin P$. Denote $f := a_0x_0 + \cdots + a_{n-1}x_{n-1} + b$, where $a_j := \bar{a}_j$ for $j \in P$, $a_j := -\bar{a}_j$ for $j \notin P$, and $b := \bar{b} + \sum_{j \notin P} \bar{a}_j$. Then f is in the pattern $\mathbf{c}$, and for any $\mathbf{v} \in \mathbb{F}_2^n$, we have $f(\mathbf{v}) = \bar{f}(\mathbf{v} \oplus \mathbf{c})$.*

Proof. Since $\bar{f}$ is in the pattern $\mathbf{0}$, we have $\bar{a}_j \geq 0$ for $0 \leq j < n$. So $a_j = \bar{a}_j \geq 0$ for $j \in P$ and $a_j = -\bar{a}_j \leq 0$ for $j \notin P$, which means f is in the pattern $\mathbf{c}$.

For any $\mathbf{v} = (v_0, \ldots, v_{n-1}) \in \mathbb{F}_2^n$, we have

$$\begin{aligned}
\bar{f}(\mathbf{v} \oplus \mathbf{c}) &= \bar{a}_0(v_0 \oplus c_0) + \cdots + \bar{a}_{n-1}(v_{n-1} \oplus c_{n-1}) + \bar{b} \\
&= \textstyle\sum_{j \in P} \bar{a}_j v_j + \sum_{j \notin P} \bar{a}_j(1 - v_j) + \bar{b} \\
&= \textstyle\sum_{j \in P} \bar{a}_j v_j + \sum_{j \notin P} (-\bar{a}_j) v_j + \sum_{j \notin P} \bar{a}_j + \bar{b} \\
&= \textstyle\sum_{j \in P} a_j v_j + \sum_{j \notin P} a_j v_j + b \\
&= f(\mathbf{v}),
\end{aligned}$$

where the second equality follows from that fact that $v_j \oplus c_j = v_j$ for $j \in P$ and $v_j \oplus c_j = 1 - v_j$ for $j \notin P$. □

Corollary 1. *Given a set $S \subset \mathbb{F}_2^n$, to compute a polynomial f in the pattern $\mathbf{c}$ such that $S \subset \mathrm{Sol}(f \geq 0)$, it suffices to compute a polynomial $\bar{f}$ in the pattern $\mathbf{0}$ such that $\{\mathbf{v} \oplus \mathbf{c} \mid \mathbf{v} \in S\} \subset \mathrm{Sol}(\bar{f} \geq 0)$.*

Proof. Assume $\bar{f}$ is *any* polynomial in the pattern $\mathbf{0}$ such that $\{\mathbf{v} \oplus \mathbf{c} \mid \mathbf{v} \in S\} \subset \mathrm{Sol}(\bar{f} \geq 0)$, then we have $\bar{f}(\mathbf{v} \oplus \mathbf{c}) \geq 0$ for any $\mathbf{v} \in S$. Let f be the polynomial defined in Prop. 1, then f is in the pattern $\mathbf{c}$, and for any $\mathbf{v} \in S$, we have $f(\mathbf{v}) = \bar{f}(\mathbf{v} \oplus \mathbf{c}) \geq 0$. □

Similarly, we have the following two corollaries.

Corollary 2. *Given a set $N \subset \mathbb{F}_2^n$, to compute a polynomial f in the pattern $\mathbf{c}$ such that $N = \mathrm{Sol}(f = 0)$, it suffices to compute a polynomial $\bar{f}$ in the pattern $\mathbf{0}$ such that $\{\mathbf{v} \oplus \mathbf{c} \mid \mathbf{v} \in N\} = \mathrm{Sol}(\bar{f} = 0)$.*

Corollary 3. *Given a set $E \subset \mathbb{F}_2^n$, to compute a polynomial f in the pattern $\mathbf{c}$ such that $E = \mathrm{Sol}(f < 0)$, it suffices to compute a polynomial $\bar{f}$ in the pattern $\mathbf{0}$ such that $\{\mathbf{v} \oplus \mathbf{c} \mid \mathbf{v} \in E\} = \mathrm{Sol}(\bar{f} < 0)$.*

By the above corollaries, we can see that computing a polynomial in the pattern $\mathbf{c}$ can always be converted to computing a polynomial in the pattern $\mathbf{0}$. All we need to do is to transform related point $\mathbf{v}$ to $\mathbf{v} \oplus \mathbf{c}$. Thus, it suffices to study the polynomials in the pattern $\mathbf{0}$ in the rest of this section. For convenience, if a polynomial f is in the pattern $\mathbf{0}$, we say it is in the **normal form**.

3.2 Properties of points in $\mathbb{F}_2^n$ with respect to a set $\bar{S}$

By Cor. 1, given a set $S \subset \mathbb{F}_2^n$, to compute a polynomial f in a pattern $\mathbf{c}$ such that $S \subset \mathrm{Sol}(f \geq 0)$, it suffices to calculate a polynomial $\bar{f}$ in the normal form such that $\{\mathbf{v} \oplus \mathbf{c} \mid \mathbf{v} \in S\} \subset \mathrm{Sol}(\bar{f} \geq 0)$. Denote $\bar{S} := \{\mathbf{v} \oplus \mathbf{c} \mid \mathbf{v} \in S\}$, before discussing how to compute $\bar{f}$ using the method of undetermined coefficients, we study the properties of the points in $\bar{S}$ and $\mathbb{F}_2^n \setminus \bar{S}$ first. The properties of these points will significantly reduce the number of constraints that are used in the computation.

We still use Example 1 to illustrate the related conceptions. To compute a polynomial in the pattern $\mathbf{c} = 0011$ with respect to the set $S = \{1000, 1010, 0110, 1110, 1001, 0101, 1101\} \subset \mathbb{F}_2^4$, it suffices to compute a polynomial $\bar{f} = \bar{a}_0x_0 + \cdots + \bar{a}_3x_3 + \bar{b}$ in the normal form such that $\bar{S} \subset \mathrm{Sol}(\bar{f} \geq 0)$ where $\bar{S} = \{1011, 1001, 0101, 1101, 1010, 0110, 1110\}$. To ensure that the condition $\bar{S} \subset \mathrm{Sol}(\bar{f} \geq 0)$ holds, the undetermined coefficients $\bar{a}_0, \bar{a}_1, \bar{a}_2, \bar{a}_3$ and constant term $\bar{b}$ must satisfy the following constraints:

$$\begin{cases} \bar{f}(1011) = & \bar{a}_0 + \bar{a}_2 + \bar{a}_3 + \bar{b} & \geq 0, \\ \bar{f}(1001) = & \bar{a}_0 + \bar{a}_3 + \bar{b} & \geq 0, \\ \bar{f}(0101) = & \bar{a}_1 + \bar{a}_3 + \bar{b} & \geq 0, \\ \bar{f}(1101) = & \bar{a}_0 + \bar{a}_1 + \bar{a}_3 + \bar{b} & \geq 0, \\ \bar{f}(1010) = & \bar{a}_0 + \bar{a}_2 + \bar{b} & \geq 0, \\ \bar{f}(0110) = & \bar{a}_1 + \bar{a}_2 + \bar{b} & \geq 0, \\ \bar{f}(1110) = & \bar{a}_0 + \bar{a}_1 + \bar{a}_2 + \bar{b} & \geq 0. \end{cases} \tag{4}$$

Note that, since $\bar{a}_j \geq 0$ for $0 \leq j < n$, some constraints in (4) can be implied by others, e.g. $\bar{f}(1011) \geq 0$ is implied by $\bar{f}(1001) \geq 0$ and $\bar{f}(1110) \geq 0$ is implied by $\bar{f}(1010) \geq 0$. Similarly, if we require the set $\mathrm{Sol}(\bar{f} < 0)$ is as large as possible, we need the following constraints to hold as many as possible:

$$\begin{array}{lcl} \bar{f}(0011) = & \bar{a}_2 + \bar{a}_3 + \bar{b} & < 0, \\ \bar{f}(0111) = & \bar{a}_1 + \bar{a}_2 + \bar{a}_3 + \bar{b} & < 0, \\ \bar{f}(1111) = & \bar{a}_0 + \bar{a}_1 + \bar{a}_2 + \bar{a}_3 + \bar{b} & < 0, \\ \bar{f}(0001) = & \bar{a}_3 + \bar{b} & < 0, \\ \bar{f}(0010) = & \bar{a}_2 + \bar{b} & < 0, \\ \bar{f}(0000) = & \bar{b} & < 0, \\ \bar{f}(1000) = & \bar{a}_0 + \bar{b} & < 0, \\ \bar{f}(0100) = & \bar{a}_1 + \bar{b} & < 0, \\ \bar{f}(1100) = & \bar{a}_0 + \bar{a}_1 + \bar{b} & < 0. \end{array}$$

However, we can now determine the constraint $\bar{f}(1111) < 0$ cannot hold, because we have $\bar{f}(1011) \geq 0$ and $\bar{a}_1 \geq 0$. That is, although the point $1111 \in \mathbb{F}_2^n \setminus \bar{S}$, we must have $1111 \in \mathrm{Sol}(\bar{f} \geq 0)$ if we require $\bar{S} \subset \mathrm{Sol}(\bar{f} \geq 0)$.

The above example shows that the properties of points in S and those of points in $\mathbb{F}_2^n \setminus S$ are different. To study the points, we need the following partial ordering and theorem.

Definition 2. For two points $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1}), \mathbf{u} = (u_0, u_1, \ldots, u_{n-1}) \in \mathbb{F}_2^n$, we say $\mathbf{v} \preceq \mathbf{u}$ if $v_i \leq u_i$ for $0 \leq i < n$. Particularly, if $\mathbf{v} \preceq \mathbf{u}$ and $\mathbf{v} \neq \mathbf{u}$, we write $\mathbf{v} \prec \mathbf{u}$.

Theorem 1. *Let $\bar{f} = \bar{a}_0 x_0 + \cdots + \bar{a}_{n-1} x_{n-1} + \bar{b}$ be a polynomial in the normal form, and $\mathbf{v}, \mathbf{u} \in \mathbb{F}_2^n$ be two points such that $\mathbf{v} \preceq \mathbf{u}$. Then we have $\bar{f}(\mathbf{v}) \leq \bar{f}(\mathbf{u})$. Moreover, $\bar{f}(\mathbf{v}) \geq 0$ implies $\bar{f}(\mathbf{u}) \geq 0$, and $\bar{f}(\mathbf{u}) < 0$ implies $\bar{f}(\mathbf{v}) < 0$.*

The proof of theorem is straightforward since $\bar{a}_j \geq 0$ for $0 \leq j < n$. Next, we give the definitions of some special points in $\mathbb{F}_2^n$ w.r.t. $\bar{S}$.

Definition 3. Let $\bar{S}$ be a subset of $\mathbb{F}_2^n$.

1. For a point $\mathbf{v} \in \mathbb{F}_2^n \setminus \bar{S}$, we say $\mathbf{v}$ is a **region** point w.r.t. $\bar{S}$, if there does not exist $\mathbf{u} \in \bar{S}$, such that $\mathbf{u} \preceq \mathbf{v}$.

2. For a point $\mathbf{v} \in \bar{S}$, we say $\mathbf{v}$ is a **border** point w.r.t. $\bar{S}$, if there exists a *region* point $\mathbf{u}$ w.r.t. $\bar{S}$, such that $\mathbf{u} \prec \mathbf{v}$ and $\mathrm{wt}(\mathbf{u}) = \mathrm{wt}(\mathbf{v}) - 1$, where $\mathrm{wt}(\mathbf{u})$ and $\mathrm{wt}(\mathbf{v})$ are the Hamming weights of $\mathbf{u}$ and $\mathbf{v}$.

3. For a point $\mathbf{v} \in \bar{S}$, we say $\mathbf{v}$ is a **minimal border** point w.r.t $\bar{S}$, if there does not exist another *border* point $\mathbf{u}$ w.r.t. $\bar{S}$, such that $\mathbf{u} \prec \mathbf{v}$.

Please note that region, border, and minimal border points are all defined with respect to the set $\bar{S}$. But for simplicity, we usually omit "w.r.t. $\bar{S}$" if there is no confusion about the set $\bar{S}$.

Consider the set $\bar{S} = \{1011, 1001, 0101, 1101, 1010, 0110, 1110\}$, which is transformed from the set S in Example 1 by XORing the point $\mathbf{c} = 0011$. All points in $\bar{S}$ are border points, and 1001, 0101, 1010, 0110 are minimal border points. Region points are 0011, 0001, 0010, 0000, 1000, 0100, and 1100.

Region points and minimal border points have the following properties.

Proposition 2. *If $\mathbf{v} \in \mathbb{F}_2^n \setminus \bar{S}$ and $\mathbf{v}$ is not a region point, $\exists \mathbf{u} \in \bar{S}$ such that $\mathbf{u} \prec \mathbf{v}$.*

Proposition 3. *If $\mathbf{v} \in \bar{S}$, there exists a minimal border point $\mathbf{u} \in \bar{S}$, such that $\mathbf{u} \preceq \mathbf{v}$.*

Corollary 4. *If $\mathbf{v} \in \mathbb{F}_2^n$ is not a region point, then there exists a minimal border point $\mathbf{u} \in \bar{S}$, such that $\mathbf{u} \preceq \mathbf{v}$. Therefore, for any polynomial $\bar{f}$ in the normal form such that $\bar{S} \subset \mathrm{Sol}(\bar{f} \geq 0)$, we have $\bar{f}(\mathbf{v}) \geq 0$.*

Corollary 5. *Let $\bar{f}$ be a polynomial in the normal form such that $\bar{S} \subset \mathrm{Sol}(\bar{f} \geq 0)$, then all points in $\mathrm{Sol}(\bar{f} < 0)$ are region points.*

Corollary 6. *Given a set $S \subset \mathbb{F}_2^n$ and a point $\mathbf{c} \in \mathbb{F}_2^n$, if $\mathbf{c} \in S$, then for any polynomial f in the pattern $\mathbf{c}$ such that $S \subset \mathrm{Sol}(f \geq 0)$, we have $\mathrm{Sol}(f < 0) = \emptyset$.*

Proof. If $\mathbf{0} \in \bar{S}$, then $\mathbf{0}$ is the only one minimal border point w.r.t. $\bar{S}$ and there is no region points w.r.t. $\bar{S}$ at all. By Cor. 5, for any polynomial $\bar{f}$ in the normal form such that $\bar{S} \subset \mathrm{Sol}(\bar{f} \geq 0)$, we have $\mathrm{Sol}(\bar{f} < 0) = \emptyset$. Then this proposition is proved by Cor. 1. □

Continue the above example, as 1001, 0101, 1010, 0110 are minimal border points. This implies only the constraints $\bar{f}(1001) \geq 0$, $\bar{f}(0101) \geq 0$, $\bar{f}(1010) \geq 0$, and $\bar{f}(0110) \geq 0$ are necessary in (4). Since 0111 and 1111 are not region points, so the constraints $\bar{f}(0111) < 0$ and $\bar{f}(1111) < 0$ cannot hold if the constraints in (4) are met.

Given a set $\bar{S}$, Alg. 1 computes the region, border, and minimal border points.

3.3 Computing a polynomial $\bar{f}$ in the normal form

Our final goal is to compute a set of inequalities to model a Boolean function F, such that the set of common solutions to these inequalities are exactly the set $\mathrm{Supp}(F)$. As we will consider the sizes and strengths of the descriptive models, the size and strength factors of the inequalities should also be considered when they are generated. According to the Sasaki-Todo algorithm, an inequality $f \geq 0$ is preferred if the set $\mathrm{Sol}(f < 0)$ is large, because this kind of inequalities helps to reduce the sizes of descriptive models. So the size factor of the inequality is related to the set $\mathrm{Sol}(f < 0)$. By the definition of strength index, the strength factor of the inequality $f \geq 0$ is related to the set $\mathrm{Sol}(f = 0)$.

To make our approaches more understandable, we first discuss how to generate an inequality by only considering its size factor. By Cor. 1, we only need to focus on the problem: given a set $\bar{S}$, how to compute a polynomial $\bar{f}$ in the normal form, such that $\bar{S} \subset \mathrm{Sol}(\bar{f} \geq 0)$ and the set $\mathrm{Sol}(\bar{f} < 0)$ is as large as possible.

Again, we consider the set $\bar{S} = \{1011, 1001, 0101, 1101, 1010, 0110, 1110\}$, which is transformed from the set S in Example 1 by XORing the point $\mathbf{c} = 0011$. In the last subsection, we know *miniBorder* $= \{1001, 0101, 1010, 0110\}$ consists of all minimal border points, and *Region* $= \{0011, 0001, 0010, 0000, 1000, 0100, 1100\}$ contains all region points. To make the constraints in $\{\bar{f}(\mathbf{v}) < 0 \mid \mathbf{v} \in \textit{Region}\}$ hold as many as possible, we introduce an auxiliary binary variable, say $y_{\mathbf{v}}$, to determine when the constraint $\bar{f}(\mathbf{v}) < 0$

Algorithm 1: Region&Border

```
   Input: $\bar{S}$: A set of points in $\mathbb{F}_2^n$.
   Output: The sets of region, border, and minimal border points.
1  begin
2      if $\mathbf{0} \in \bar{S}$ then
3          return $\emptyset$, $\{\mathbf{0}\}$, $\{\mathbf{0}\}$
4      $Region, Border, miniBorder \longleftarrow \{\mathbf{0}\}, \emptyset, \emptyset$
5      $weight[i] \longleftarrow \{\mathbf{v} \in \mathbb{F}_2^n \mid \mathrm{wt}(\mathbf{v}) = i\}$, for $0 \le i \le n$
6      for $i$ from $1$ to $n$ do
7          for $\mathbf{v} \in weight[i]$ do
8              $D \longleftarrow \{\mathbf{u} \mid \mathbf{u} \prec \mathbf{v}\} \cap weight[i-1]$
9              if $\mathbf{v} \in \bar{S}$ then
10                 if $\exists\, \mathbf{u} \in D$ s.t. $\mathbf{u} \in Region$ then
11                     $Border \longleftarrow Border \cup \{\mathbf{v}\}$
12                 if $\forall\, \mathbf{u} \in D$ s.t. $\mathbf{u} \in Region$ then
13                     $miniBorder \longleftarrow miniBorder \cup \{\mathbf{v}\}$
14             else
15                 if $\forall\, \mathbf{u} \in D$ s.t. $\mathbf{u} \in Region$ then
16                     $Region \longleftarrow Region \cup \{\mathbf{v}\}$
17     return $Region$, $Border$, $miniBorder$
```

holds. Next, we use the following MILP model to solve for the coefficients of $\bar{f}$, where $\bar{f}(\mathbf{v})$ has the form of $\bar{a}_0x_0 + \bar{a}_1x_1 + \bar{a}_2x_2 + \bar{a}_3x_3 + \bar{b}$:

$$\text{Maximize}: \sum_{\mathbf{v} \in Region} y_{\mathbf{v}}, \tag{5}$$

$$s.t. \begin{cases} \bar{f}(\mathbf{v}) \ge 0, & \mathbf{v} \in miniBorder, \\ \bar{f}(\mathbf{v}) - \gamma \cdot (1 - y_{\mathbf{v}}) < 0, & \mathbf{v} \in Region, \end{cases}$$

where γ is a positive integer and its value is discussed below.

Please note that in the above MILP model, $\bar{a}_j$'s are all unknown positive integers and $\bar{b}$ is an unknown integer, and they have lower and upper bounds in an MILP solver. For $\bar{a}_0, \bar{a}_1, \bar{a}_2, \bar{a}_3$, we assume they are not bigger than a positive integer R, i.e. $\bar{a}_j \in [0, R]$, which means $0 \le \bar{a}_j \le R$. Here, the constant R is usually chosen as a multiple of the dimension $n = 4$, e.g. $R = 25n$. With this setting, the value of $\bar{a}_0x_0 + \bar{a}_1x_1 + \bar{a}_2x_2 + \bar{a}_3x_3$ ranges from 0 to $4R$, so the range of $\bar{b}$ is set as $[-4R, 0]$. Because if $\bar{b} < -4R$ or $\bar{b} > 0$, we always have $\bar{f}(\mathbf{v}) < 0$ and $\bar{f}(\mathbf{v}) > 0$ for all $\mathbf{v} \in \mathbb{F}_2^4$. Therefore, the range of the values of $\bar{f}$ is $[-4R, 4R]$. Then the constant γ is often set as $4R + 1$, such that if $y_{\mathbf{v}} = 1$, the constraint $\bar{f}(\mathbf{v}) - \gamma \cdot (1 - y_{\mathbf{v}}) = \bar{f}(\mathbf{v}) < 0$ works; otherwise, $\bar{f}(\mathbf{v}) - \gamma \cdot (1 - y_{\mathbf{v}}) = \bar{f}(\mathbf{v}) - (4R + 1) < 0$ always holds and hence does not constrain the values of $\bar{a}_j$'s and $\bar{b}$.

After solving the MILP model (5), the optimal solution leads to a polynomial $\bar{f}_0 = 2x_0 + x_1 + 3x_2 + 3x_3 - 4$. We can verify that $\bar{f}_0(\mathbf{v}) \ge 0$ for all $\mathbf{v} \in \bar{S}$, and $\mathrm{Sol}(\bar{f}_0 < 0) = \{1100, 1000, 0100, 0010, 0001, 0000\}$. However, as the model (5) does not consider the strength index of the inequality, the generated polynomial $\bar{f}_0$ is not good enough, and we only have $\mathrm{Sol}(\bar{f}_0 = 0) = \{0110, 0101\}$.

To increase the strength of an inequality $\bar{f} \ge 0$, we expect the set $\mathrm{Sol}(\bar{f} = 0)$ also contains as many border points as possible, because we believe in this case the inequality $\bar{f} \ge 0$ can eliminate many useless numerical solutions and hence leads to a better strength.

To compute a polynomial $\bar{f}$ by considering both $\mathrm{Sol}(\bar{f} < 0)$ and $\mathrm{Sol}(\bar{f} = 0) \cap \{$border points of $\bar{S}\}$, the idea is the same as that used to maximize the size of $\mathrm{Sol}(\bar{f} < 0)$. Specifically, we introduce a new auxiliary binary variable, say $z_{\mathbf{v}}$, to determine whether a border point $\mathbf{v}$ is in $\mathrm{Sol}(\bar{f} = 0)$. Assume the sets *Region*, *Border*, and *miniBorder* contain all region, border, and minimal border points w.r.t. $\bar{S}$. We use the following MILP model to compute the desired polynomials:

$$\text{Maximize}: \alpha\Big(\sum_{\mathbf{v}\in Region} y_{\mathbf{v}}\Big) + \beta\Big(\sum_{\mathbf{v}\in Border} z_{\mathbf{v}}\Big), \tag{6}$$

$$s.t. \begin{cases} \bar{f}(\mathbf{v}) \geq 0, & \mathbf{v} \in miniBorder, \\ \bar{f}(\mathbf{v}) - \gamma \cdot (1 - y_{\mathbf{v}}) < 0, & \mathbf{v} \in Region, \\ \bar{f}(\mathbf{v}) - \gamma \cdot (1 - z_{\mathbf{v}}) \leq 0, & \mathbf{v} \in Border, \end{cases}$$

where γ is a positive integer. And as discussed in the above, we usually choose $\gamma = nR + 1$ and R is the upper bound of the coefficients in $\bar{f}$.

In the model (6), we have two parameters, α and β, to adjust the weights of points in $\mathrm{Sol}(\bar{f} < 0)$ and $\mathrm{Sol}(\bar{f} = 0) \cap Border$. For example, if we set $\alpha = 2^n$ and $\beta = 1$, then the above model will first try to find the polynomials $\bar{f}$'s such that $\mathrm{Sol}(\bar{f} < 0)$ is maximal, and then among these $\bar{f}$'s, the model finds the one such that the set $\mathrm{Sol}(\bar{f} = 0) \cap Border$ is maximal. Note that the MILP model (5) is a specialization of the model (6) when $\alpha = 1$ and $\beta = 0$.

Let us consider the set $\bar{S} = \{1011, 1001, 0101, 1101, 1010, 0110, 1110\}$ again. We know $Region = \{0011, 0001, 0010, 0000, 1000, 0100, 1100\}$, $Border = \bar{S}$, and $miniBorder = \{1001, 0101, 1010, 0110\}$. By solving model (5), we obtain a polynomial $\bar{f}_0 = 2x_0 + x_1 + 3x_2 + 3x_3 - 4$, and $\mathrm{Sol}(\bar{f}_0 < 0) = \{1100, 1000, 0100, 0010, 0001, 0000\}$. But the set $\mathrm{Sol}(\bar{f}_0 = 0)$ only contains two border points, 0110 and 0101. Solving the MILP model (6) by setting $\alpha = 2^4$ and $\beta = 1$, we get the polynomial $\bar{f}_0' = x_0 + x_1 + 2x_2 + 2x_3 - 3$. For this $\bar{f}_0'$, we have $\mathrm{Sol}(\bar{f}_0' < 0) = \mathrm{Sol}(\bar{f}_0 < 0)$ and $\mathrm{Sol}(\bar{f}_0' = 0) \cap Border = \{1010, 0110, 1001, 0101\}$, which means the inequalities $\bar{f}_0' \geq 0$ and $\bar{f}_0 \geq 0$ have the same size factor, but $\bar{f}_0' \geq 0$ has a larger strength index.

Computing irredundant polynomials: By adjusting the values of α and β in the model (6), we can generate many polynomials, but we need to prevent the model from computing the redundant polynomials. We define a polynomial $\bar{f}$ is **redundant** if there exists another polynomial $\bar{f}'$ such that $\mathrm{Sol}(\bar{f} < 0) \subset \mathrm{Sol}(\bar{f}' < 0)$ and $\mathrm{Sol}(\bar{f} = 0) \subset \mathrm{Sol}(\bar{f}' = 0)$.

With the above example, to compute another polynomial that is irrendundant to $\bar{f}_0' = x_0 + x_1 + 2x_2 + 2x_3 - 3$, we can append the constraint : $y_{0011} + z_{1011} + z_{1101} + z_{1110} \geq 1$ to the MILP model (6), since $y_{0011} = 0$ and $z_{1011} = z_{1101} = z_{1110} = 0$ holds in the solution corresponding to $\bar{f}_0'$. By solving this new MILP model, we obtain $\bar{f}_1' = 5x_0 + 5x_1 - x_2 - x_3 - 6$, and $\mathrm{Sol}(\bar{f}_1' < 0) = \{0011, 1000, 0100, 0010, 0001, 0000\} \not\subset \mathrm{Sol}(\bar{f}_0' < 0)$, which means $\bar{f}_1'$ is irredundant to $\bar{f}_0'$.

In fact, by appending new constraints to the model (6) iteratively, we can compute *almost all* irredundant polynomials in theory. Please note that we say "almost all" instead of "all", because we have the following conjecture but we cannot prove it at present.

Conjecture 1. *There exists a positive constant R in $\mathbb{Z}$, such that for any polynomial $\bar{f}$, there always exists a polynomial $\bar{f}'$ s.t.* $\mathrm{Sol}(\bar{f} < 0) = \mathrm{Sol}(\bar{f}' < 0)$, $\mathrm{Sol}(\bar{f} = 0) = \mathrm{Sol}(\bar{f}' = 0)$, *and the absolute values of the coefficients in $\bar{f}'$ are not bigger than R.*

The following algorithm computes all irredundant polynomials if Conj. 1 is true.

Algorithm 2: IrredundantPolynomials

```
Input: $\bar{S}$: a set of points in $\mathbb{F}_2^n$; $\alpha$, $\beta$: constants.
Output: The set of all irredundant polynomials $\bar{f}$'s such that $\bar{S} \subset \mathrm{Sol}(\bar{f} \geq 0)$.
1  begin
2      $Region, Border, miniBorder \longleftarrow$ Region&Border($\bar{S}$)
3      Prepare an empty MILP model $\mathcal{M}$
4      $\mathcal{M}.var \longleftarrow \bar{a}_j$ as an integer variables with the range $[0, R]$, for $0 \leq j < n$
5      $\mathcal{M}.var \longleftarrow \bar{b}$ as an integer variable with the range $[-nR, 0]$
6      $\mathcal{M}.var \longleftarrow y_{\mathbf{v}}$ as a binary variable, for $\mathbf{v} \in Region$
7      $\mathcal{M}.var \longleftarrow z_{\mathbf{v}}$ as a binary variable, for $\mathbf{v} \in Border$
8      $\mathcal{M}.con \longleftarrow \bar{f}(\mathbf{v}) \geq 0$, for $\mathbf{v} \in miniBorder$
9      $\mathcal{M}.con \longleftarrow \bar{f}(\mathbf{v}) - (nR+1) \cdot (1 - y_{\mathbf{v}}) \leq -1$, for $\mathbf{v} \in Region$
10     $\mathcal{M}.con \longleftarrow \bar{f}(\mathbf{v}) - (nR+1) \cdot (1 - z_{\mathbf{v}}) \leq 0$, for $\mathbf{v} \in Border$
11     $\mathcal{M}.obj \longleftarrow maximize(\alpha(\sum_{\mathbf{v} \in Region} y_{\mathbf{v}}) + \beta(\sum_{\mathbf{v} \in Border} z_{\mathbf{v}}))$
12     $Result \longleftarrow \emptyset$
13     $\mathcal{M}.optimize()$
14     while $\mathcal{M}$ is not infeasible do
15         Let $\{\tilde{a}_j$'s, $\tilde{b}$, $\tilde{y}_{\mathbf{v}}$'s, $\tilde{z}_{\mathbf{v}}$'s$\}$ be an optimal solution of $\mathcal{M}$
16         $Result \longleftarrow Result \cup \{\tilde{a}_0 x_0 + \cdots + \tilde{a}_{n-1} x_{n-1} + \tilde{b}\}$
17         $\mathcal{M}.con \longleftarrow \sum_{\tilde{y}_{\mathbf{v}}=0} y_{\mathbf{v}} + \sum_{\tilde{z}_{\mathbf{v}}=0} z_{\mathbf{v}} \geq 1$
18         $\mathcal{M}.optimize()$
19     return $Result$
```

We briefly talk about the complexity of Alg. 2. The complexity of Alg. 2 contains two parts. The first part is the complexity of computing one polynomial. This complexity is usually determined by the number of constraints added in Line 8 $\sim$ 10. According to our experiments, the size of the set *Region* dominates this complexity. That is, the larger the set *Region* is, the longer time it takes to compute a polynomial. The second part is the number of possible irredundant polynomials, which is usually determined by the size of $Region \cup Border$. Generally, if the dimension n is large, e.g. $n \geq 12$, it takes too long to obtain all irredundant polynomials. So in this case, we often stop the algorithm after a number of polynomials have been obtained.

Computing other special polynomials: More special polynomials can also be generated by adding corresponding constraints to the model (6). For example, we can even generate the sparsest polynomials by using auxiliary binary variables to determine whether coefficients are 0's. Specifically, assume $\bar{a}_j$ is a coefficient ranging from 0 to R. Let w_j be a binary variable, then the following constraints could determine whether $\bar{a}_j$ is 0:

$$\bar{a}_j \geq 1 - w_j \text{ and } \bar{a}_j \leq R \cdot (1 - w_j).$$

Note that if $w_j = 1$, we must have $\bar{a}_j = 0$; otherwise, we have $1 \leq \bar{a}_j \leq R$. So by adding $\sum w_j$ to the objective function, we can control the number of nonzero coefficients in $\bar{f}$.

4 Generating a descriptive model of a Boolean function

Generally, a descriptive model is used as a component of automatic search programs. Different descriptive models of the same Boolean function often lead to distinct efficiencies of automatic search programs.

Given a set of candidate inequalities, Sasaki and Todo proposed an algorithm to select inequalities such that the number of selected inequalities is minimal in [ST17a], i.e. the Sasaki-Todo algorithm generates a descriptive model with the minimal size. This algorithm was also used in the follow-up research, including [BC20, Udo21]. However, it is found that the descriptive model with the minimal size does not always lead to the best efficiency of the automatic search programs. For this phenomenon, we think there are two possible reasons. Firstly, the inequalities in consideration are not good enough, because before this paper, the kinds of inequalities that can be generated were quite limited. Secondly, the Sasaki-Todo algorithm only considers the *size* of the descriptive models but ignores the influence from the *strength* of the descriptive models.

In [Vie15], the author pointed out that the descriptive models which balance the size and strength usually lead to better efficiencies. We agree with this viewpoint, but think it is not easy to apply this viewpoint to practical computations, because the strengths of the descriptive models are hard to be calculated when we are selecting the inequalities. So we need an approximate concept to evaluate the strengths of descriptive models.

Given a Boolean function F, the convex hull of $\mathrm{Supp}(F)$ has the largest strength [Hoj21]. Intuitively, one important characteristic of the convex hull is that, every point on the "surface" of $\mathrm{Supp}(F)$ often lies on many hyperplanes of the convex hull. We plan to use this feature to define the approximate strength of a descriptive model. Note that the point on the "surface" is in fact a border point with respect to some set, and a point $\mathbf{v}$ lying on a hyperplane defined by the inequality $f \geq 0$ equals to $\mathbf{v} \in \mathrm{Sol}(f = 0)$. Then we have the following definition.

Definition 4. Let F be a Boolean function from $\mathbb{F}_2^n$ to $\mathbb{F}_2$, and $\bar{S}_{\mathbf{c}} := \{\mathbf{v} \oplus \mathbf{c} \mid \mathbf{v} \in \mathrm{Supp}(F)\}$ for $\mathbf{c} \in \mathbb{F}_2^n$. Denote B by the set of all border points w.r.t. $\bar{S}_{\mathbf{c}}$, i.e.

$$B := \bigcup_{\mathbf{c} \in \mathbb{F}_2^n} \{\text{border points w.r.t. } \bar{S}_{\mathbf{c}}\}.$$

Let $D = \{f_0 \geq 0, f_1 \geq 0, \ldots, f_m \geq 0\}$ be a set of inequalities that models F, we define the **approximate strength** of D as

$$\min_{\mathbf{v} \in B} |\{f_i \mid \mathbf{v} \in \mathrm{Sol}(f_i = 0), 0 \leq i \leq m\}|,$$

where $|\cdot|$ means the cardinality of a set. Particularly, to differentiate the descriptive models whose approximate strengths are 0's, we define the **cover rate** of D as

$$|(\cup_{0 \leq i \leq m} \mathrm{Sol}(f_i = 0)) \cap B| / |B|.$$

Please note that, for a point $\mathbf{v}$, the number $|\{f_i \mid \mathbf{v} \in \mathrm{Sol}(f_i = 0), 0 \leq i \leq m\}|$ is actually the number of hyperplanes that $\mathbf{v}$ lies on. Besides, if the approximate strength of D is not 0, then its cover rate is surely 1. So the cover rate is mainly used to compare the strengths of descriptive models whose approximate strengths are 0's. With conception of the approximate strength, then we can classify the descriptive models into different types according to their approximate strengths.

Since B is the set of all border points, we have $B \subseteq \mathrm{Supp}(F)$ according to Definition 3. Actually, we always have $B = \mathrm{Supp}(F)$ in our experiments. Besides, we generated many sets of points by random, and computed their convex hulls. The approximate strengths of these convex hulls are all large. So we think the definition of approximate strength is reasonable.

Next, we present an algorithm (Alg. 3) to compute a descriptive model by balancing the size and the approximate strength. This algorithm improves the Sasaki-Todo algorithm, and computes a descriptive model with a minimal size among the descriptive models with an approximate strength being at least s. When $s = 0$, this algorithm computes a descriptive model with the minimal size, and the cover rate of this descriptive model is maximal in this size.

The constraint in Line 9 ensures the approximate strength of the output descriptive model is at least s. In case $s = 0$, the objective function $2^n \cdot (\sum_{0 \leq i \leq m} d_i) - \sum_{\mathbf{v} \in B} c_{\mathbf{v}}$

guarantees the output descriptive models must have the minimal size first, and then guides the solver to find a descriptive model with a relatively high cover rate $(\sum_{\mathbf{v}\in B} c_{\mathbf{v}})/|B|$.

```
Algorithm 3: BalanceSize&Strength
Input: $F$: a Boolean function; $P = \{f_0 \geq 0, f_1 \geq 0, \dots, f_m \geq 0\}$: a set of
       candidate inequalities s.t. $\mathrm{Supp}(F) \subset \mathrm{Sol}(f_i \geq 0)$ for $0 \leq i \leq m$; $s$: an
       approximate strength.
Output: $D$: a descriptive model of $F$, with the minimal size such that the
        approximate strength of $D$ is at least $s$.
1  begin
2      $B \longleftarrow \bigcup_{\mathbf{c}\in\mathbb{F}_2^n}\{$border points w.r.t. $\bar{S}_{\mathbf{c}}\}$, where $\bar{S}_{\mathbf{c}} = \{\mathbf{v} \oplus \mathbf{c} \mid \mathbf{v} \in \mathrm{Supp}(F)\}$
3      Prepare an empty MILP model $\mathcal{M}$
4      $\mathcal{M}.var \longleftarrow d_i$ as a binary variable, for $0 \leq i \leq m$
5      $\mathcal{M}.var \longleftarrow c_{\mathbf{v}}$ as a binary variable, for $\mathbf{v} \in B$
6      for $\mathbf{v} \in \mathbb{F}_2^n \setminus \mathrm{Supp}(F)$ do
7          $\mathcal{M}.con \longleftarrow \sum_{\mathbf{v}\in\mathrm{Sol}(f_i<0)} d_i \geq 1$
8      for $\mathbf{v} \in B$ do
9          $\mathcal{M}.con \longleftarrow \sum_{\mathbf{v}\in\mathrm{Sol}(f_i=0)} d_i \geq s$
10         $\mathcal{M}.con \longleftarrow \sum_{\mathbf{v}\in\mathrm{Sol}(f_i=0)} d_i \geq c_{\mathbf{v}}$
11     $\mathcal{M}.obj \longleftarrow minimize(2^n \cdot (\sum_{0\leq i\leq m} d_i) - \sum_{\mathbf{v}\in B} c_{\mathbf{v}})$
12     $\mathcal{M}.optimize()$
13     return $\{f_i \geq 0 \mid d_i = 1\}$
```

Next, we compute two descriptive models for the Boolean function in Example 1. The support of the Boolean function is $S = \{1000, 1010, 0110, 1110, 1001, 0101, 1101\} \subset \mathbb{F}_2^4$.

To generate the candidate inequalities, parameters are set as $\alpha = 2^4$, $\beta = 1$, and $R = 100$ in Alg. 2, and polynomials in all 2^4 patterns are computed. The following inequalities are the results.

$$\begin{aligned}
\text{pattern } 0000: &\quad 100x_0 + 99x_1 + x_2 + x_3 - 100 \geq 0,\\
&\quad x_0 + x_1 - 1 \geq 0,\\
\text{pattern } 0100: &\quad x_0 - 99x_1 + 100x_2 + 100x_3 - 1 \geq 0,\\
&\quad -x_1 + x_2 + x_3 \geq 0,\\
\text{pattern } 1100: &\quad -x_1 + x_2 + x_3 \geq 0,\\
\text{pattern } 0010: &\quad x_0 + x_1 - 1 \geq 0,\\
\text{pattern } 0001: &\quad x_0 + x_1 - 1 \geq 0,\\
\text{pattern } 0011: &\quad x_0 + x_1 - 2x_2 - 2x_3 + 1 \geq 0,\\
&\quad 5x_0 + 5x_1 - x_2 - x_3 - 4 \geq 0,\\
&\quad -x_2 - x_3 + 1 \geq 0,\\
&\quad x_0 + x_1 - 1 \geq 0,\\
\text{pattern } 1011: &\quad -x_2 - x_3 + 1 \geq 0,\\
\text{pattern } 0111: &\quad -x_2 - x_3 + 1 \geq 0,\\
\text{pattern } 1111: &\quad -x_2 - x_3 + 1 \geq 0.
\end{aligned}$$

Note that there are no inequalities for the pattern $\mathbf{v} \in S$, because for any inequality $f \geq 0$ in the pattern $\mathbf{v} \in S$, we have $\mathrm{Sol}(f < 0) = \emptyset$. Besides, some inequalities belong to several patterns, e.g. $-x_2 - x_3 + 1 \geq$ are in the patterns 0011, 1011, 0111, and 1111, because its coefficients of x_0 and x_1 are zeros. Thus, the candidate inequalities are $P = \{100x_0 + 99x_1 + x_2 + x_3 - 100 \geq 0, x_0 + x_1 - 1 \geq 0, x_0 - 99x_1 + 100x_2 + 100x_3 - 1 \geq 0, -x_1 + x_2 + x_3 \geq 0, x_0 + x_1 - 2x_2 - 2x_3 + 1 \geq 0, 5x_0 + 5x_1 - x_2 - x_3 - 4 \geq 0, -x_2 - x_3 + 1 \geq 0\}$.

Setting $s = 0$ in Alg. 3, the algorithm computes a descriptive model with the minimal size and a relatively large cover rate. The result consists of the following two inequalities:

$$\begin{aligned} x_0 + x_1 - 2x_2 - 2x_3 + 1 &\geq 0, \\ x_0 - 99x_1 + 100x_2 + 100x_3 - 1 &\geq 0. \end{aligned}$$

Note that in this example, we have $B = S$. But $\mathrm{Sol}(x_0+x_1-2x_2-2x_3+1=0) \cup \mathrm{Sol}(x_0 - 99x_1 + 100x_2 + 100x_3 - 1 = 0) = \{1001, 0101, 1010, 0110, 1000\} \neq S$. So the approximate strength of this descriptive model is 0, and the cover rate is 5/7.

By setting $s = 1$ in Alg. 3, the output descriptive model contains three inequalities:

$$\begin{aligned} 5x_0 + 5x_1 - x_2 - x_3 - 4 &\geq 0, \\ x_0 + x_1 - 2x_2 - 2x_3 + 1 &\geq 0, \\ -x_1 + x_2 + x_3 &\geq 0. \end{aligned}$$

From the above example, we can see that the approximate strengths of the descriptive models with the minimal sizes tend to be 0, and to get larger approximate strengths, more inequalities are needed.

5 Experiments

We designed two types of experiments. The first type of experiment aims to test the diversity of the inequalities generated by the SuperBall approach. In the second type of experiment, we generated several types of descriptive models of different Boolean functions, trying to find out which type of descriptive model leads to a better efficiency.

5.1 Testing the diversity of inequalities

By saying a set of inequalities, say $P = \{f_0 \geq 0, f_1 \geq 0, \ldots, f_m \geq 0\}$ has a good diversity, we tried to mean that the number of irredundant inequalities in P is large, and the solutions of the functions f_i's are different from each other. To reflect the diversity of a set of inequalities, we have the following definition.

Definition 5. Let F be a Boolean function from $\mathbb{F}_2^n$ to $\mathbb{F}_2$ and $P_F = \{f_0 \geq 0, f_1 \geq 0, \ldots, f_m \geq 0\}$ be a set of inequalities which describes F. We define the **diversity index** of P_F as

$$|\{(\mathbf{v}_1, \mathbf{v}_2) \in \mathbb{F}_2^n \setminus \mathrm{Supp}(F) \mid \exists f \geq 0 \in P_F \text{ s.t. } f(\mathbf{v}_1) < 0 \text{ and } f(\mathbf{v}_2) < 0\}|,$$

where $\mathrm{Supp}(F)$ is the support of F and $|\cdot|$ is the cardinality of a set.

We say P_F has a good "diversity", if its diversity index is large.

We compared the diversity indexes of the sets of inequalities generated by the SuperBall approach with those generated by the Convex hull method [SHW+14b] or Espresso method [AST+17]. In practice, there are off-the-shelf softwares to implement the Convex hull method [SHW+14b] or Espresso method, such as Sage and Logic Friday. So we generated the inequalities by Sage and Logic Friday, then computed the diversity indexes and the minimal sizes of the descriptive models of Sboxes. Since the method in [SHW+14b] could not be applied to large Sboxes, we only consider the Boolean functions deduced from the 4-bit Sboxes.

The results in Table 1 show that the diversities of the sets of inequalities generated by our approach are greater than those of [SHW+14b] and [AST+17]. Table 1 also lists the maximum values of the diversity index (max D.I.) for each Sbox. For a Boolean function F, we compute its maximum value of the diversity index as follows. For any pair $(\mathbf{v}_1, \mathbf{v}_2) \in \mathbb{F}_2^n \setminus \mathrm{Supp}(F)$, we construct the MILP model via the undetermined coefficients method to verify whether there exists an inequality $f \geq 0$ such that $f(\mathbf{v}_1) < 0$ and $f(\mathbf{v}_2) < 0$. The coefficients of the inequality have the same ranges as Alg. 2. The number of pairs $(\mathbf{v}_1, \mathbf{v}_2)$ that $\mathbf{v}_1$ and $\mathbf{v}_2$ can be rejected by one inequality simultaneously is the maximum value of the diversity index. For the 4-bit Sbox in Table 1, the diversity index of the set of inequalities generated by the SuperBall approach has reached its maximum value. Besides, we also computed the minimal descriptive models and found that the size of the minimal descriptive models could reflect the diversity of the set of candidate inequalities. That is, for a Boolean function, if its minimal descriptive model has a smaller size, the corresponding set of candidate inequalities always has a greater diversity.

Table 1: Comparisons of diversity indexes (D.I.) of sets of candidate inequalities and the minimal size of descriptive models. Boolean functions are deduced from Sboxes.

	Sage		LogicalFriday		Our results		Max D.I.
	D.I.	Size	D.I.	Size	D.I.	Size	
Present [BKL+07]	1489	21	226	38	**2007**	16	2007
Klein [GSL11]	1072	21	191	43	**1565**	18	1565
Twine [KSMM12]	1175	23	224	47	**1621**	19	1621
Prince [BCG+12]	1245	22	270	53	**1644**	18	1644
Piccolo [SIH+11]	1711	21	759	31	**2764**	14	2764
MIBS [ISSK09]	1094	23	161	52	**1437**	20	1437
LBlock S0-1,5-8 [WZ11]	2008	24	586	30	**2898**	15	2898
LBlock S2-4,9 [WZ11]	2008	24	603	31	**2898**	15	2898
Serpent S0 [BAK98]	1436	21	227	37	**1940**	16	1940
Serpent S1 [BAK98]	1414	21	286	38	**1896**	16	1896
Serpent S2 [BAK98]	1574	21	233	43	**2019**	17	2019
Serpent S3 [BAK98]	1227	27	162	49	**1572**	20	1572
Serpent S4 [BAK98]	1257	23	207	43	**1601**	18	1601
Serpent S5 [BAK98]	1358	23	231	42	**1729**	18	1729
Serpent S6 [BAK98]	1393	21	296	41	**1936**	16	1936
Serpent S7 [BAK98]	1148	27	292	41	**1483**	20	1483
Liliput [ATPC+19]	1185	23	232	47	**1636**	19	1636
Minalpher [YYK+14]	1162	22	232	52	**1634**	18	1634
Midori S0 [BBI+15]	1521	21	459	48	**2098**	16	2098
Midori S1 [BBI+15]	1429	22	306	57	**1771**	20	1771
Rectangle [ZBL+14]	1708	21	302	31	**2303**	15	2303
Skinny-64 [BJK+16]	1711	21	759	31	**2764**	14	2764
Gift [BPP+17]	1664	21	332	34	**2284**	16	2284
Pride [ADK+14]	1955	21	796	31	**3087**	16	3087

Since the sets of the candidate inequalities in [BC20] and [Udo21] are not available, the diversities can not be computed directly. Our way of testing the diversity of inequalities generated by the methods of [BC20] and [Udo21] is as follows. Firstly, we generate three kinds of inequalities via Alg. 2[2] by setting $(\alpha = 2^n, \beta = 1)$, $(\alpha = 1, \beta = 2^n)$, and $(\alpha = 1, \beta = 1)$, and gather all inequalities. Then, we compute a descriptive model with the minimal size via the Sasaki-Todo algorithm. Thirdly, we compare the sizes of descriptive models with existing best results, e.g. results in [BC20, Udo21]. As the authors in [BC20, Udo21] also used the Sasaki-Todo algorithm to obtain the minimal sizes, the differences between the minimal sizes would reflect the diversities of the set of candidate inequalities. Specifically, the smaller the size is, the greater diversity of inequalities is,

[2]The number of irredundant polynomials in a pattern is huge if the dimension n is large, so we stop the algorithm when $3 \sim 5$ polynomials in the pattern are obtained.

which is demonstrated by the results in Table 1.

In the experiments, we considered the Boolean functions deduced from Sboxes, linear layers, and the division property. The comparisons of minimal descriptive models from different Sboxes are shown in Table 2. Generally, if an Sbox has more invalid differential propagations, the number of region points of the Sbox is larger. That is because the region consists of invalid differential propagations. For example, the block cipher SKINNY-128 has 54067 invalid differential propagations through an Sbox, while other 8-bit Sboxes have only 33150, so the maximal number of region points of SKINNY-128 is abnormally higher compared to others. As the number of region points in a pattern reflects the difficulty of computing inequalities, we also list the maximal number of region points among the 2^n patterns in the table. Comparisons of minimal descriptive models from different linear layers and the division properties are shown in Table 3.

From Table 2 and 3, we can see that the sizes of our descriptive models are always the smallest. Udovenko's method found the minimal sizes of descriptive models for many Sboxes, but his method failed for the Sboxes of Keccak , Fides-6, SC2000-6, and AES. We think the main reason is that, Udovenko's method can only compute one kind of inequalities, i.e. his algorithm is equivalent to Alg. 2 by setting $\alpha = 2^n$ and $\beta = 0$. But we generated three kinds of inequalities. So the diversity of our inequalities is better than Udovenko's. Besides, we think the efficiency of Udovenko's method also limits its performance.

In all, the experimental results show that the inequalities generated by the SuperBall approach have a great diversity.

5.2 Finding out the most efficient descriptive models

To find out the most efficient descriptive models, we considered the Boolean functions deduced from the Sboxes of three block ciphers, Liliput, SKINNY-128, AES. We prepared 4 types of descriptive models generated by Alg. 3 as well as 4 counterparts from existing works. To check the efficiency, we designed two traditional search tasks for each cipher.

Specifically, the descriptive models in consideration were generated as follows.

Balance-s: These types of descriptive models were generated in the following two steps. Firstly, three kinds of inequalities were generated using Alg. 2[3] by setting $(\alpha = 2^n, \beta = 1)$, $(\alpha = 1, \beta = 2^n)$, and $(\alpha = 1, \beta = 1)$. Secondly, Alg. 3 was called to compute the descriptive models by setting the approximate strength bound as s. Particularly, in case $s = 0$, the descriptive model output by Alg. 3 also has the maximal cover rate, so we denote the descriptive model of this type as Balance-0-crmax.

Balance-0-cr0: The descriptive model of this type was modified[4] from the corresponding descriptive model Balance-0-crmax. For each inequality $f \geq 0$ in Balance-0-crmax, we computed a new polynomial f', such that $\mathrm{Sol}(f' < 0) = \mathrm{Sol}(f < 0)$ and $\mathrm{Sol}(f' = 0) = \emptyset$. Thus, the size of Balance-0-cr0 is the same as that of Balance-0-crmax, but there is no border points lying on the hyperplanes of the inequalities of Balance-0-cr0, which means the cover rate of Balance-0-cr0 is 0. So Balance-0-crmax always has a slightly larger strength than Balance-0-cr0. By considering this type of descriptive model, we hope to see whether the strength really affects the efficiency.

Sage+ST: Descriptive models of this kind were constructed by the algorithm in [ST17a]. The candidate inequalities were generated by Sage [Dev16], and then the Sasaki-Todo algorithm was applied to find the descriptive models with the minimal sizes.

Sage+Greedy: Descriptive models of this kind were constructed by the algorithm in [SHW$^+$14a]. The candidate inequalities were generated by Sage, and then a greedy method was used to reduce the sizes of descriptive models.

[3]The algorithm is stoped when $3 \sim 5$ polynomials are obtained in each pattern.

[4]This modification is also done by the method of undetermined coefficients, by converting the desired requests to constraints on the coefficients and constant terms.

Table 2: Comparisons of the minimal sizes of descriptive models. Boolean functions are deduced from Sboxes.

Sbox	Max $\lvert Region \rvert$	[BC20]	[Udo21]	Our results
	4-bit Sboxes ($n = 8$)			
PRESENT [BKL+07]	30	17	**16**	**16**
KLEIN [GSL11]	27	19	**18**	**18**
TWINE [KSMM12]	28	**19**	**19**	**19**
PRINCE [BCG+12]	40	19	**18**	**18**
PICCOLO [SIH+11]	49	16	**14**	**14**
MIBS [ISSK09]	26	**20**	**20**	**20**
LBLOCK † [WZ11]	55	17	-	**15**
SERPENT S0 [BAK98]	30	17	-	**16**
SERPENT S1 [BAK98]	30	17	-	**16**
SERPENT S2 [BAK98]	33	18	-	**17**
SERPENT S3 [BAK98]	24	**20**	-	**20**
SERPENT S4 [BAK98]	32	19	-	**18**
SERPENT S5 [BAK98]	32	19	-	**18**
SERPENT S6 [BAK98]	30	17	-	**16**
SERPENT S7 [BAK98]	24	**20**	-	**20**
LILIPUT [ATPC+19]	28	**19**	-	**19**
MINALPHER [YYK+14]	25	19	-	**18**
MIDORI S0 [BBI+15]	51	**16**	**16**	**16**
MIDORI S1 [BBI+15]	35	**20**	**20**	**20**
RECTANGLE [ZBL+14]	39	17	**15**	**15**
SKINNY-64 [BJK+16]	49	16	**14**	**14**
GIFT [BPP+17]	36	17	**16**	**16**
PRIDE [ADK+14]	43	**16**	**16**	**16**
	5-bit Sboxes ($n = 10$)			
KECCAK [BDPA11]	231	34	-	**26**
ASCON [DEMS21]	139	32	**27**	**27**
FIDES-5 [BBK+13]	74	61	**57**	**57**
SC2000-5 [SYY+01]	68	64	**60**	**60**
	6-bit Sboxes ($n = 12$)			
APN-6 [BDMW10]	183	167	**145**	**145**
FIDES-6 [BBK+13]	123	180	166	**165**
SC2000-6 [SYY+01]	112	214	205	**202**
	7-bit Sboxes ($n = 14$)			
WG-7 [LCGL10]	350	-	-	**562**
MISTY S7 [Mat97]	187	-	-	**693**
WAGE [MRG+19]	246	-	-	**588**
	8-bit Sboxes ($n = 16$)			
AES [DR05]	356	2882	2699	**2425**
SKINNY-128 [BJK+16]	20362	302	-	**177**
ZUC [Tea18]	354	-	-	**2417**
ARIA S2 [KKP+03]	372	-	-	**2427**

† LBLOCK represents for LBLOCK S0∼9 .

Sage: The descriptive model of this kind consists of all inequalities generated by Sage. So the descriptive model is in fact the convex hull, and has the largest strengths, as well

Table 3: Comparisons of the minimal sizes of descriptive models. Boolean functions are deduced from linear layers and division properties.

Cipher	Dim.	Max $\lvert Region \rvert$	[BC20]	[XZBL16]	[HSWW20]	Our results
Linear layers						
Midori [BBI+15]	8	73	32			**8**
Skinny-128 [BJK+16]	8	125	18			**6**
Minalpher [YYK+14]	8	73	32			**8**
Craft [BLMR19]	8	132	16			**8**
Two-subset division property						
Simon [BSS+13]	8	145		6*		**3**
Trivium [DCP08]	10	664		7*		**6**
Three-subset division property without unknown subset						
Simon [BSS+13]	8	175			10*	**3**
Trivium [DCP08]	10	781			13*	**4**

* The inequalities involve auxiliary variables.

as large sizes.

Espresso: Descriptive models of this kind were constructed by the algorithm in [AST+17]. The authors used the Espresso algorithm implemented by Logic Friday (http://sontrak.com/) to generate linear inequalities for the DDT of an Sbox. They translated the problem of searching for inequalities into the classical problem of minimization of the product-of-sum representation of Boolean functions, which is a well-studied problem, and Espresso algorithm is a heuristic method to solve this problem.

The sizes and strengths of the above descriptive models are shown in Table 4. Please note that the numbers in the table are all related to one single Sbox. If k Sboxes are used in a cipher, the numbers of constraints per round in the search model should be k times of the corresponding numbers in the table.

Table 4: The sizes (Size), approximate strengths (A.S.), and cover rates (C.Rate) of descriptive models (for a single Sbox) of Liliput, `SKINNY`, and AES.

	Liliput			`SKINNY`			AES		
	Size	A.S.	C.Rate	Size	A.S.	C.Rate	Size	A.S.	C.Rate
Balance-0-cr0	19	0	0	177	0	0	2425	0	0
Balance-0-crmax	19	0	93/106	177	0	6420/11469	2425	0	15584/32386
Balance-1	22	1	1	189	1	1	3423	1	1
Balance-3	34	3	1	367	3	1	8806	3	1
Sage+ST	23	0	101/106	-	-	-	-	-	-
Sage+Greedy	26	0	102/106	-	-	-	-	-	-
Sage	324	12	1	-	-	-	-	-	-
Espresso	-	-	-	377	4	1	8310	1	1

Two traditional search tasks were prepared as follows.

Verification of differential pairs: For an r-round cipher, this task is to check whether a given input and output difference pair $(\delta_{in}, \delta_{out})$ is possible, i.e. whether there exists a differential trail connecting δ_{in} and δ_{out}. In each search, the program is stopped if one differential trail is found or the model is proved infeasible. For Liliput and `SKINNY`, we generated 1000 random input and output pairs, and took the average time. For round-reduced AES algorithm, we only considered 250 random input and output pairs, since each search of AES took more time than the other two ciphers.

Finding the minimal number of active Sboxes: For an r-round cipher, this task is to find out the minimal number of active Sboxes in feasible differential trails. In this task, there are no constraints on the input and output differential pairs. For each round-reduced cipher, we conducted 5 searches and took the average time. Since this task costs too much computing time for the cipher AES, we only report the times for the ciphers Liliput and SKINNY-128.

As we only want to compare the performances of different types of descriptive models for the Sboxes, the constraints of the other components, e.g. the linear layers, are all the same in the experiments. The comparisons are shown in Table 5, 6, and 7. Our platform is AMD Threaddripper 3990x 2.9GHz, 256 GB RAM, running Ubuntu 20.04. The tables also show the numbers of rounds, the numbers of Sboxes in total, and the total numbers of inequalities related to these Sboxes.

Table 5: Comparisons on Liliput. There are 16 Sboxes in each round.

	Verification of Differential pairs			
	28 rounds (448 Sboxes)		30 rounds (480 Sboxes)	
	Time(sec)	#Ineq.	Time(sec)	#Ineq.
Balance-0-cr0	7.75	8512	9.24	9120
Balance-0-crmax	**6.38**	8512	**7.56**	9120
Balance-1	6.87	9856	8.35	10560
Balance-3	7.82	15232	9.97	16320
Sage+ST	7.68	10304	9.02	11040
Sage+Greedy	7.19	11648	8.61	12480
Sage	24.73	145152	26.97	155520
	Finding least active Sboxes			
	4 rounds (64 Sboxes)		6 rounds (96 Sboxes)	
	Time(sec)	#Ineq.	Time(sec)	#Ineq.
Balance-0-cr0	39.62	1216	696.16	1824
Balance-0-crmax	**8.22**	1216	**535.81**	1824
Balance-1	13.93	1408	758.58	2112
Balance-3	10.85	2176	2330.98	3264
Sage+ST	13.58	1472	741.67	2208
Sage+Greedy	14.72	1664	854.73	2496
Sage	71.02	20736	1603.77	31104

From the experimental results in Table 5, 6, and 7, we have three observations.

Firstly, the strengths of descriptive models affect the efficiency. Note that although many descriptive models have the same minimal size, the strengths of them are still different, while the differences are reflected on the cover rates. This is why we design the comparisons between "Balance-0-crmax" and "Balance-0-cr0". By definition, the cover rate of Balance-0-cr0 is 0, but Balance-0-crmax usually has a high cover rate. The experimental results show that the descriptive model Balance-0-crmax always has a better efficiency than Balance-0-cr0. We think this fact is reasonable, because there are fewer useless numerical solutions to the descriptive model Balance-0-crmax than those of Balance-0-cr0, the MILP solver should have fewer obstacles to find out the useful integer solutions.

Secondly, although there is no type of descriptive model that always has the best efficiency, but the descriptive model Balance-0-crmax, which has the minimal size and a relatively high cover rate, has a better efficiency in most of experiments, particularly when the total number of Sboxes is large. In our opinion, although the descriptive models with larger strengths contain fewer useless numerical solutions, the sizes of the descriptive

Table 6: Comparisons on SKINNY-128. There are 16 Sboxes in each round.

	Verification of Differential pairs			
	30 rounds (480 Sboxes)		32 rounds (512 Sboxes)	
	Time(sec)	#ineq.	Time(sec)	#ineq.
Balance-0-cr0	26.61	84960	27.61	90624
Balance-0-crmax	**24.61**	84960	**26.17**	90624
Balance-1	31.35	90720	33.56	96768
Balance-3	87.51	174720	115.73	187904
Espresso	93.41	180960	129.46	193024
	Finding least active Sboxes			
	3 rounds (48 Sboxes)		5 rounds (80 Sboxes)	
	Time(sec)	#ineq.	Time(sec)	#ineq.
Balance-0-cr0	61.54	8496	>6000	14160
Balance-0-crmax	59.94	8496	>6000	14160
Balance-1	**14.87**	9072	>6000	15120
Balance-3	178.35	17616	**2170.58**	29360
Espresso	87.22	18096	2556.91	30160

Table 7: Comparisons on AES. There are 16 Sboxes in each round.

	Verification of Differential pairs			
	3 rounds (48 Sboxes)		5 rounds (80 Sboxes)	
	Time(sec)	#ineq.	Time(sec)	#ineq.
Balance-0-cr0	115.30	116400	571.61	194000
Balance-0-crmax	103.26	116400	**511.36**	194000
Balance-1	**100.68**	164304	731.20	273840
Balance-3	363.49	422688	2268.45	704480
Espresso	331.75	398880	1874.49	664800

models play a more important role in the solving procedure. Specifically, to solve an MILP model, the solvers always finds a numerical solution to the model first, and then searches for integer solutions around the numerical one. We think solving of an MILP model that has a larger strength may save some time in the second phase. But the matrix operations are the fundamental operations in both phases, and the sizes of the matrices are directly determined by the sizes of the descriptive models. So the sizes of descriptive models often affect the efficiency more. The descriptive model Balance-0-crmax has the smallest size and a not bad strength, so we think this is why it performs better in most experiments.

Thirdly, if the difference between the sizes is not large, the descriptive models that have larger approximate strengths may have better efficiencies. This phenomenon appears in the task of finding least active Sboxes of SKINNY-128.

Thus, in all, we suggest the type of descriptive model that has the minimal size and relatively high cover rate. Although this type of descriptive model does not achieve the best efficiency all the time, it has a relatively stable performance.

6 Conclusions

In this paper, we propose the SuperBall approach to generate various inequalities. Since the method of undetermined coefficients is used, we are able to generate any kinds of inequalities by adding appropriate constraints in theory. However, not all inequalities are useful for constructing efficient descriptive models. We focus on the inequalities that could affect the sizes and strengths of descriptive models. Specifically, we only consider the inequality $f \geq 0$ such that $\mathrm{Sol}(f < 0)$ and $\mathrm{Sol}(f = 0)$ are large. But we cannot guarantee the other kinds of inequalities are useless.

Our initial goal is to find a type of descriptive model such that the models of this type always have the best efficiencies. We think this goal is partially reached. We found one type of descriptive model, and the models of this type have better performances in most of our experiments. But we should admit that descriptive models of this type are not always the most efficient. Thus, for other ciphers that are not considered in this paper, we suggest using the descriptive model that has the minimal size and a relatively high cover rate first. If the efficiency of this descriptive model is not good enough, one could try to improve this descriptive model by generating other kinds of inequalities by the SuperBall approach.

Acknowledgments

We would like to thank Yunwen Liu and the anonymous reviewers for their constructive comments. This work was supported by the National Natural Science Foundation of China (Grant No. 61877058, Grant No. 61977060, Grant No. 62002354).

References

[ADK+14] Martin R. Albrecht, Benedikt Driessen, Elif Bilge Kavun, Gregor Leander, Christof Paar, and Tolga Yalçın. Block Ciphers – Focus on the Linear Layer (feat. PRIDE). In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014*, pages 57–76, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[AST+17] 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 Trans. Symmetric Cryptol.*, 2017(4):99–129, 2017.

[ATPC+19] Adomnicai Alexandre, Berger Thierry P., Clavier Christophe, Francq Julien, Huynh Paul, Lallemand Virginie, Gouguec Kevin Le, Minier Marine, Reynaud Leo, and Thomas Gaël. Lilliput-AE: a New Lightweight Tweakable Block Cipher for Authenticated Encryption with Associated Data. 2019. https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/LILLIPUT-AE-spec.pdf.

[BAK98] Eli Biham, Ross Anderson, and Lars Knudsen. Serpent: A New Block Cipher Proposal. In Serge Vaudenay, editor, *Fast Software Encryption*, pages 222–238, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[BBI+15] Subhadeep Banik, Andrey Bogdanov, Takanori Isobe, Kyoji Shibutani, Harunaga Hiwatari, Toru Akishita, and Francesco Regazzoni. Midori: A Block Cipher for Low Energy. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology – ASIACRYPT 2015*, pages 411–436, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[BBK+13] Begül Bilgin, Andrey Bogdanov, Miroslav Knežević, Florian Mendel, and Qingju Wang. Fides: Lightweight Authenticated Cipher with Side-Channel Resistance for Constrained Hardware. In Guido Bertoni and Jean-Sébastien Coron, editors, *Cryptographic Hardware and Embedded Systems - CHES 2013*, pages 142–158, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[BC20] Christina Boura and Daniel Coggia. Efficient MILP Modelings for Sboxes and Linear Layers of SPN ciphers. *IACR Trans. Symmetric Cryptol.*, 2020(3):327–361, 2020.

[BCG$^+$12] Julia Borghoff, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knezevic, Lars R. Knudsen, Gregor Leander, Ventzislav Nikov, Christof Paar, Christian Rechberger, Peter Rombouts, Søren S. Thomsen, and Tolga Yalçın. PRINCE – A Low-Latency Block Cipher for Pervasive Computing Applications. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, pages 208–225, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[BDMW10] K.A. Browning, J.F. Dillon, M.T. McQuistan, and A.J. Wolfe. An APN Permutation in Dimension Six. *Postproceedings of the 9th International Conference on Finite Fields and Their Applications Fq'9*, 518, 01 2010.

[BDPA11] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. The Keccak reference, Version 3.0. In *https://keccak.team/keccak.html*, 2011.

[BJK$^+$16] Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. The SKINNY Family of Block Ciphers and Its Low-Latency Variant MANTIS. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016*, pages 123–153, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[BKL$^+$07] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems - CHES 2007*, pages 450–466, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[BKS09] Julia Borghoff, Lars Ramkilde Knudsen, and Mathias Stolpe. Bivium as a Mixed Integer Programming Problem. In Matthew G. Parker, editor, *Cryptography and Coding*, number 5921 in Lecture Notes of Computer Science, pages 133–152. Springer, 2009. 12th IMA International Conference on Cryptography and Coding, IMACC 2009 ; Conference date: 15-12-2009 Through 17-12-2009.

[BLMR19] Christof Beierle, Gregor Leander, Amir Moradi, and Shahram Rasoolzadeh. CRAFT: Lightweight Tweakable Block Cipher with Efficient Protection Against DFA Attacks. Cryptology ePrint Archive, Report 2019/210, 2019. https://ia.cr/2019/210.

[BPP$^+$17] Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. GIFT: A Small Present. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems – CHES 2017*, pages 321–345, Cham, 2017. Springer International Publishing.

[BSS$^+$13] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The SIMON and SPECK Families of Lightweight Block Ciphers. Cryptology ePrint Archive, Report 2013/404, 2013. https://ia.cr/2013/404.

[DCP08] Christophe De Cannière and Bart Preneel. *Trivium*, pages 244–266. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[DEMS21] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2: Lightweight Authenticated Encryption and Hashing. *Journal of Cryptology*, 34, 07 2021.

[Dev16] The Developers. SageMath, the Sage Mathematics Software System. 01 2016.

[DR05] Joan Daemen and Vincent Rijmen. *Rijndael/AES*, pages 520–524. Springer US, Boston, MA, 2005.

[FWG+16] Kai Fu, Meiqin Wang, Yinghua Guo, Siwei Sun, and Lei Hu. MILP-Based Automatic Search Algorithms for Differential and Linear Trails for Speck. In Thomas Peyrin, editor, *Fast Software Encryption*, pages 268–288, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[GSL11] Zheng Gong, Nikova Svetla, and Yee Wei Law. KLEIN: A new family of lightweight block ciphers. volume 7055, pages 1–18, 06 2011.

[HLM+20] Yonglin Hao, Gregor Leander, Willi Meier, Yosuke Todo, and Qingju Wang. Modeling for Three-Subset Division Property Without Unknown Subset. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020*, pages 466–495, Cham, 2020. Springer International Publishing.

[Hoj21] Christopher Hojny. Strong IP formulations need large coefficients. *Discrete Optimization*, 39:100624, 2021.

[HSWW20] Kai Hu, Siwei Sun, Meiqin Wang, and Qingju Wang. An Algebraic Formulation of the Division Property: Revisiting Degree Evaluations, Cube Attacks, and Key-Independent Sums. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2020*, pages 446–476, Cham, 2020. Springer International Publishing.

[HW18] Kai Hu and Meiqin Wang. Automatic Search for A Variant of Division Property Using Three Subsets (Full Version). Cryptology ePrint Archive, Report 2018/1187, 2018. https://ia.cr/2018/1187.

[ISSK09] Maryam Izadi, Babak Sadeghiyan, Seyed Saeed Sadeghian, and Hossein Arabnezhad Khanooki. MIBS: A New Lightweight Block Cipher. In Juan A. Garay, Atsuko Miyaji, and Akira Otsuka, editors, *Cryptology and Network Security*, pages 334–348, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[KKP+03] Daesung Kwon, Jaesung Kim, Sangwoo Park, Soo Sung, Yaekwon Sohn, Junghwan Song, Yongjin Yeom, E-Joong Yoon, Sangjin Lee, Jaewon Lee, Seongtaek Chee, Daewan Han, and Jin Hong. New block cipher: ARIA. pages 432–445, 11 2003.

[KSMM12] Eita Kobayashi, Tomoyasu Suzaki, Kazuhiko Minematsu, and Sumio Morioka. TWINE: A Lightweight Block Cipher for Multiple Platforms. volume 7707, 03 2012.

[LCGL10] Yiyuan Luo, Qi Chai, Guang Gong, and Xuejia Lai. A Lightweight Stream Cipher WG-7 for RFID Encryption and Authentication. In *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, pages 1–6, 2010.

[Mat97] Mitsuru Matsui. New block encryption algorithm MISTY. In Eli Biham, editor, *Fast Software Encryption*, pages 54–68, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.

[MELR13] Germán Morales-España, Jesus M. Latorre, and Andres Ramos. Tight and Compact MILP Formulation for the Thermal Unit Commitment Problem. *IEEE Transactions on Power Systems*, 28(4):4897–4908, 2013.

[MRG+19] Aagaard Mark, AlTawy Riham, Gong Guang, Mandal Kalikinkar, Rohit Raghvendra, and Zidaric Nusa. WAGE: An Authenticated Cipher. Submission to the NIST LWC Competition, 2019. https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/spec-doc-rnd2/wage-spec-round2.pdf.

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming. In *Proceedings of the 7th International Conference on Information Security and Cryptology*, Inscrypt'11, page 57–76, Berlin, Heidelberg, 2011. Springer-Verlag.

[SHW+14a] Siwei Sun, Lei Hu, Meiqin Wang, Peng Wang, Kexin Qiao, Xiaoshuang Ma, Danping Shi, Ling Song, and Kai Fu. Towards Finding the Best Characteristics of Some Bit-oriented Block Ciphers and Automatic Enumeration of (Related-key) Differential and Linear Characteristics with Predefined Properties. Cryptology ePrint Archive, Report 2014/747, 2014. https://ia.cr/2014/747.

[SHW+14b] Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma, and Ling Song. Automatic Security Evaluation and (Related-key) Differential Characteristic Search: Application to SIMON, PRESENT, LBlock, DES(L) and Other Bit-Oriented Block Ciphers. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014*, pages 158–178, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[SIH+11] Kyoji Shibutani, Takanori Isobe, Harunaga Hiwatari, Atsushi Mitsuda, Toru Akishita, and Taizo Shirai. Piccolo: An Ultra-Lightweight Blockcipher. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2011*, pages 342–357, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[ST17a] Yu Sasaki and Yosuke Todo. New Algorithm for Modeling S-box in MILP Based Differential and Division Trail Search. In Pooya Farshim and Emil Simion, editors, *Innovative Security Solutions for Information Technology and Communications*, pages 150–165, Cham, 2017. Springer International Publishing.

[ST17b] Yu Sasaki and Yosuke Todo. New Impossible Differential Search Tool from Design and Cryptanalysis Aspects. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017*, pages 185–215, Cham, 2017. Springer International Publishing.

[Sun21] Yao Sun. Towards the Least Inequalities for Describing a Subset in Z_2^n. Cryptology ePrint Archive, Paper 2021/1084, 2021. https://eprint.iacr.org/2021/1084.

[SWW17] Ling Sun, Wei Wang, and Meiqin Wang. Automatic Search of Bit-Based Division Property for ARX Ciphers and Word-Based Division Property. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 128–157, Cham, 2017. Springer International Publishing.

[SYY$^+$01] Takeshi Shimoyama, Hitoshi Yanami, Kazuhiro Yokoyama, Masahiko Takenaka, Kouichi Itoh, Jun Yajima, Naoya Torii, and Hidema Tanaka. The block cipher SC2000. pages 312–327, 04 2001.

[Tea18] The ZUC Team. The ZUC-256 Stream Cipher, 2018. http://www.is.cas.cn/ztzl2016/zouchongzhi/201801/W020180126529970733243.pdf.

[TIHM18] 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.

[Udo21] Aleksei Udovenko. MILP modeling of Boolean functions by minimum number of inequalities. Cryptology ePrint Archive, Report 2021/1099, 2021. https://ia.cr/2021/1099.

[Vie15] Juan Pablo Vielma. Mixed integer linear programming formulation techniques. *SIAM Review*, 57(1):3–57, 2015.

[WHG$^+$19] Senpeng Wang, Bin Hu, Jie Guan, Kai Zhang, and Tairong Shi. MILP-aided Method of Searching Division Property Using Three Subsets and Applications. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019*, pages 398–427, Cham, 2019. Springer International Publishing.

[WW11] Shengbao Wu and Mingsheng Wang. Security Evaluation against Differential Cryptanalysis for Block Cipher Structures. *IACR Cryptology ePrint Archive*, 2011:551, 01 2011.

[WZ11] Wenling Wu and Lei Zhang. LBlock: A Lightweight Block Cipher. In Javier Lopez and Gene Tsudik, editors, *Applied Cryptography and Network Security*, pages 327–344, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[XZBL16] 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 Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016*, pages 648–678, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[YYK$^+$14] Sasaki Yu, Todo Yosuke, Aoki Kazumaro, Naito Yusuke, Sugawara Takeshi, Murakami Yumiko, Matsui Mitsuru, and Shoichi Hirose. Minalpher v1. 2014. https://competitions.cr.yp.to/round1/minalpherv1.pdf.

[ZBL$^+$14] Wentao Zhang, Zhenzhen Bao, Dongdai Lin, Vincent Rijmen, Bohan Yang, and Ingrid Verbauwhede. RECTANGLE: A Bit-slice Lightweight Block Cipher Suitable for Multiple Platforms. Cryptology ePrint Archive, Report 2014/084, 2014. https://ia.cr/2014/084.

[ZZDX20] Chunning Zhou, Wentao Zhang, Tianyou Ding, and Zejun Xiang. Improving the MILP-based Security Evaluation Algorithm against Differential/Linear Cryptanalysis Using A Divide-and-Conquer Approach. *IACR Transactions on Symmetric Cryptology*, pages 438–469, 01 2020.