Convexity of division property transitions: theory, algorithms and compact models*

Aleksei Udovenko

CryptoExperts, Paris, France
aleksei@affine.group

October 12, 2021

Abstract. Integral cryptanalysis is a powerful tool for attacking symmetric primitives, and *division property* is a state-of-the-art framework for finding integral distinguishers.

This work describes new theoretical and practical insights into traditional bit-based division property. We focus on analyzing and exploiting monotonicity/convexity of division property and its relation to the graph indicator. In particular, our investigation leads to a *new compact representation* of propagation, which allows CNF/MILP modeling for larger S-Boxes, such as 16-bit Super-Sboxes of lightweight block ciphers or even 32-bit *random* S-boxes. This solves the challenge posed by Derbez and Fouque (ToSC 2020), who questioned the possibility of SAT/SMT/MILP modeling of 16-bit Super-Sboxes. As a proof-of-concept, we model the Super-Sboxes of the 8-round LED by CNF formulas, which was not feasible by any previous approach.

Our analysis is further supported by an elegant algorithmic framework. We describe simple algorithms for computing division property of a set of n-bit vectors in time $O(n2^n)$, reducing such sets to minimal/maximal elements in time $O(n2^n)$, computing division property propagation table of an $n \times m$-bit S-box and its compact representation in time $O((n+m)2^{n+m})$. In addition, we develop an advanced algorithm tailored to "heavy" bijections, allowing to model, for example, a randomly generated 32-bit S-box.

Keywords: Division Property · S-boxes · SAT · CNF · MILP · LED

Acknowledgements

The author thanks the anonymous reviewers for their helpful comments and Claude Carlet for fruitful discussions on graph indicators and division property.

1 Introduction

With the ongoing surge of lightweight cryptography, the field of cryptanalysis of lightweight symmetric primitives is pressured to evaluate the security as precisely as possible: adding a few extra rounds as a security margin is not affordable in the lightweight setting. Among the most powerful cryptanalysis techniques are linear and differential cryptanalysis, and integral cryptanalysis. For example, the long-standing MISTY1 [Mat97] block cipher was broken recently by integral cryptanalysis [Tod15a, BK16] (based on division property, the topic of this work) with a surprisingly low time complexity 2^{70}. While provable security arguments against linear and differential cryptanalysis exist already since the design of the AES block cipher [DR02], provable security arguments against integral attacks started to appear only recently [HLLT20, HLLT21].

Division property is a state-of-the-art technique for finding integral distinguishers in symmetric ciphers. Since the seminal work of Todo [Tod15b] focusing on word/state-based division property, many improvements and variants of the technique were developed. The focus shifted towards bit-based division property [TM16a], followed by a surprisingly effective MILP-based approach [XZBL16] (mixed-integer linear programming) of finding division property-based distinguishers via the search of the so-called *division trails*. This line continued with a series of works improving MILP and SAT/SMT-based (satisfiability modulo theories) modeling [SWW16, ST, HWW20, GD21]. Classic (also called *traditional* or *conventional*) division property is *imperfect*: it may miss an integral distinguisher, although it never produces a false positive. A more recent advancement is the development of "perfect" monomial prediction techniques [HLM+20, HSWW20, HLLT20], which require *counting division trails* and so far are computationally feasible only in a few cases. This work focuses on traditional division property, as it remains powerful and the most widely applicable tool for integral cryptanalysis.

From the theory side, following preliminary analysis [SHZ+16, GRW16], the work of Boura and Canteaut [BC16] formalized and studied the *state-based* division property in terms of *parity sets*. In particular, they showed that state-based division property of a set is defined by the set's algebraic *degree*. While many of their results about parity sets translate directly into *bit-based* division property, such links were not explicitly stated. To the best of our understanding, the theory behind *bit-based* division property is not fully developed. Furthermore, very recently, Carlet [Car20a] proposed method for bounding the algebraic degree of a composition of function from the degrees of their *graph indicators*. It is a natural question whether division property can be improved by incorporating such bounds. A recent work [CXZZ21] studied formally relationships between different variants of division property and algebraic degree bounds for composite functions, such as the Boura-Canteaut bound [BC13]. However, this work did not consider graph indicator-based bounds, leaving this gap open. As a part of this work, we aim to fill the aforementioned gaps and extend the theory, focusing on the monotonicity/convexity aspects of division property and relations with the graphs of the analyzed functions.

The imperfectness of traditional bit-based division property shows up in various ways. Division property analysis can be applied to any Boolean circuit implementation of a cipher (constructed from e.g. AND and XOR gates). However, due to the imperfectness, information gets lost during propagation through the circuit. Considering larger parts of the cipher, such as S-boxes and linear maps, allows to slow down the loss of information. For example, Zhang and Rijmen [ZR18] showed that propagating division property through a linear map via a basic COPY-and-XOR implementation is imperfect. The right way to handle a linear map is to encode all invertible square submatrices of the linear map's matrix. A typical linear layer of a lightweight block cipher operates on at least 16 bits and its matrix may contain a large number of invertible submatrices. Encoding the division property propagation through such a layer in a SAT/MILP instance deemed to be not feasible until recently, when Hu, Wang and Wang [HWW20] proposed a generic

SMT-based solution, which is feasible for up to 64-bit linear maps. Lambin, Derbez and Fouque [LDF20] showed that propagation through S-boxes is also fragile: combining an S-Box with a linear map may also result in loss or gain of information.

To battle the imperfectness of traditional division property, Derbez and Fouque [DF20] proposed to increase its precision by considering a Super-Sbox - a composition of the cipher's linear map with the adjacent S-boxes - as a single propagation unit. For many lightweight block ciphers, Super-Sboxes are 16-bit bijections. The results of [DF20] shows that this approach increases precision significantly and allows to find new integral distinguishers for 1-2 more rounds for some ciphers. However, SAT/MILP modeling of Super-Sboxes was not feasible by state-of-the-art techniques and the authors of [DF20] had to develop an ad-hoc search technique. In fact, they challenged the community to develop SAT/MILP modeling of such large mappings: "*We also believe this work will challenge the community in handling such large propagation tables with generic solvers for MILP, SAT or SMT models.*". As a part of this work, we provide a solution to this challenge, based on our theoretical advancement.

Our contribution This work focuses on theory and practice of *traditional division property*. All other variants, such as three-subset division property [TM16a] (and without the unknown subset [HLM+20]), monomial prediction [HSWW20], are out of scope of this work. The main contributions of this work are:

1. Development of the theoretic framework behind the classic division property. This includes fine-grained (bit-based) formulations of previous statements about division property, exhibiting convexity of division property and its relation to the recent graph indicator-based bounds by Carlet [Car20a].

2. Compact characterization of division property propagation through a function F by means of the (reduced) division property of its graph. This yields compact constraint systems for MILP/SAT solvers, allowing us to model much larger S-boxes than was previously possible, including 16-bit Super-Sboxes and, as a proof-of-concept, randomly generated 32-bit S-boxes. We also introduce additional techniques for improving modeling efficiency.

3. A framework for manipulation of dense sets of binary vectors. It includes simple algorithms for computing division property of a set of n-bit vectors (complexity $\mathcal{O}(n2^n)$), reducing such sets to minimal/maximal elements (complexity $\mathcal{O}(n2^n)$), computing division property propagation table of an $n \times m$-bit S-box and its compact representation (complexity $\mathcal{O}((n+m)2^{n+m})$). These algorithms improve previous best algorithms by a factor of 2^n. In addition, we develop an advanced algorithm for the compact representation tailored to "heavy" n-bit bijections, for which it runs in time $\tilde{O}(2^n)$ (heuristically).

4. As a proof-of-concept, we apply our techniques to 8-round LED and show that its Super-Sbox model does not yield integral distinguishers (although they might still exist), even with linear masks applied to an input and an output Super-Sbox. This fills the gap left by [DF20], as their approach was not feasible for LED.

Our implementations are written in a mix of Python and C++, featuring performance and a convenientAPI (an excerpt is provided in Appendix A). All the source code is made publicly available. For details, see:

https://github.com/CryptoExperts/AC21-DivProp-Convexity

Outline Section 2 provides the necessary background with a focus on the partial order on bit-vectors. In Section 3, we briefly reintroduce traditional division property and develop its theory, culminating in a new compact representation. As a byproduct, we exhibit a direct link between division property and graph indicators. The following Section 4 focuses on CNF/MILP modeling aspects of the new representation. Section 5 presents our algorithmic framework for manipulating dense sets of binary vectors. Finally, in Section 6, we show how our techniques can be applied to model the Super-Sbox representation of LED.

2 Preliminaries

Boolean operations AND,OR,XOR,NOT denoted respectively by $\wedge, \vee, \oplus, \neg$ can be applied to (pairs of) single bits or bitwise to bit-vectors. We use $\underline{1} \in \mathbb{F}_2^n$ (resp. $\underline{0}$) to denote the all-one (resp. all-zero) vector of a dimension n depending on the context. We write $\neg x \coloneqq x \oplus \underline{1}$ and $\neg X \coloneqq \{\neg x \mid x \in X\}$, $X \subseteq \mathbb{F}_2^n$, to disambiguate from the set complement $\overline{X} \coloneqq \{y \in \mathbb{F}_2^n \mid y \notin X\}$. The unit vectors $e_j \in \mathbb{F}_2^n, 0 \le j < n$, are the vectors with the j-th (0-based) coordinate equal to 1 and all other coordinates equal to 0.

The notation $x^u, u \in \mathbb{F}_2^n$, is used to denote the monomial $\prod_{i=0}^{n-1} x_i^{u_i}$, letting $x_i^0 = 1$. Any Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ has a unique expression $f(x) = \bigoplus_{u \in \mathbb{F}_2^n} a_u x^u$, where $a_u \in \mathbb{F}_2$. This expression is called the *algebraic normal form* (ANF) of f. We say that f *contains* the monomial x^u if $a_u = 1$ in the ANF of f. The ANF support of f, denoted $\mathrm{Supp}_{\mathrm{ANF}}(f)$, is the set of all exponents u with $a_u = 1$ in the ANF of f.

The indicator vector of a set $X \subseteq \mathbb{F}_2^n$ is the vector $I \in \mathbb{F}_2^{2^n}$ such that $I_x = 1$ if and only if $x \in X$. Here we use the natural identification of $\mathbb{F}_2^n$ with $\{0, \dots, 2^n - 1\}$. By an abuse of notation, we will identify a set X with its indicator vector implicitly. The indicator function of X is the map $\mathbb{1}_X : \mathbb{F}_2^n \to \mathbb{F}_2 : x \mapsto I_x$.

The *graph* of a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, denoted $\mathbf{\Gamma}_F$, is the set

$$\mathbf{\Gamma}_F = \{(x, y) \mid x \in \mathbb{F}_2^n, y = F(x)\} \subseteq \mathbb{F}_2^n \times \mathbb{F}_2^m.$$

The *graph indicator* of F is the indicator function of its graph $\mathbf{\Gamma}_F$.

2.1 Partial order

We use the product order on vectors over $\mathbb{F}_2$, which is, for $x, y \in \mathbb{F}_2^n$, $x \preceq y$ if and only if $x_i \le y_i$ for all i. We write $x \prec y$ if $x \preceq y$ and $x \ne y$.

Definition 1. The *lower closure* of a set $X \subseteq \mathbb{F}_2^n$, denoted by $\mathrm{LowerClosure}(X)$, is the set of all $u \in \mathbb{F}_2^n$ with $u \preceq x$ for some $x \in X$:

$$\mathrm{LowerClosure}(X) \coloneqq \{u \in \mathbb{F}_2^n \mid \exists x \in X : u \preceq x\} = \bigcup_{x \in X} \{u \in \mathbb{F}_2^n \mid u \preceq x\}.$$

The *upper closure* of a set $X \subseteq \mathbb{F}_2^n$, denoted by $\mathrm{UpperClosure}(X)$, is the set of all $u \in \mathbb{F}_2^n$ with $x \preceq u$ for some $x \in X$:

$$\mathrm{UpperClosure}(X) \coloneqq \{u \in \mathbb{F}_2^n \mid \exists x \in X : u \succeq x\} = \bigcup_{x \in X} \{u \in \mathbb{F}_2^n \mid u \succeq x\}.$$

A set X is an *upper set* if its upper closure is X itself. A set X is a *lower set* if its lower closure is X itself.

Remark 1. An intuitive interpretation is as follows. For each vector in X, the upper closure converts positions with the value 0 into a wildcard, whereas the lower closure converts positions with the value 1 into a wildcard.

Example 1. $\text{LowerClosure}\left(\{\texttt{110}, \texttt{001}\}\right) = \{\texttt{000}, \texttt{010}, \texttt{100}, \texttt{110}, \texttt{000}, \texttt{001}\}$.

Example 2. $\text{UpperClosure}\left(\{\texttt{110}, \texttt{001}\}\right) = \{\texttt{001}, \texttt{011}, \texttt{101}, \texttt{110}, \texttt{111}\}$.

Proposition 1. *Let X, Y be lower sets (resp. upper sets). Then, $X \cup Y$ and $X \cap Y$ are lower sets (resp. upper sets); $\overline{X}$ is an upper set (resp. a lower set).*

Definition 2. A subset $X \subseteq \mathbb{F}_2^n$ is called *convex*, if for any $a, b, c \in \mathbb{F}_2^n$, $a \preceq b \preceq c$ and $a, c \in X$ imply $b \in X$. An equivalent condition is

$$X = \text{LowerClosure}\left(X\right) \cap \text{UpperClosure}\left(X\right).$$

Definition 3. The *max-set* of a set $X \subseteq \mathbb{F}_2^n$, denoted by $\text{MaxSet}\left(X\right)$, is the set of all maximal elements in X:

$$\text{MaxSet}\left(X\right) \coloneqq \{u \in X \mid \nexists x \in X : x \succ u\}.$$

The *min-set* of a set $X \subseteq \mathbb{F}_2^n$, denoted by $\text{MinSet}\left(X\right)$, is the set of all minimal elements in X:

$$\text{MinSet}\left(X\right) \coloneqq \{u \in X \mid \nexists x \in X : x \prec u\}.$$

Max-/min-sets are compact representations of lower/upper sets. Max-/min-sets are *antichains* (their elements are pairwise incomparable) and so are convex.

Proposition 2. *The operator $\neg$ anti-commutes with* MinSet*,* MaxSet*,* LowerClosure*,* UpperClosure*: for any set X,*

$$\neg\text{MinSet}\left(X\right) = \text{MaxSet}\left(\neg X\right), \qquad \neg\text{LowerClosure}\left(X\right) = \text{UpperClosure}\left(\neg X\right),$$
$$\neg\text{MaxSet}\left(X\right) = \text{MinSet}\left(\neg X\right), \qquad \neg\text{UpperClosure}\left(X\right) = \text{LowerClosure}\left(\neg X\right).$$

3 New insights into division property

We start by briefly reformulating the traditional bit-based division property in terms of parity sets in Subsection 3.1. Then, we present a complete link with the set indicator (Subsection 3.2). This link helps us to develop new characterization of transitions (Theorem 1), which in turn leads to a compact representation. Next, Subsection 3.6 summarizes the observed convex structure of division property transitions, setting the basement for modeling techniques described in Section 4. In Subsection 3.7, we revisit the approach of applying input/output linear masks and reformulate it in our framework. Finally, relationships with recent graph indicator-based degree bounds by Carlet [Car20a] are investigated in Subsection 3.8.

3.1 Division property and parity sets

Boura and Canteaut [BC16] introduced the notion of *parity sets* as another view of division property.

Definition 4 (Parity set [BC16])**.** The *parity set* of a set $X \subseteq \mathbb{F}_2^n$, denoted $\text{ParitySet}\left(X\right)$, is the set of all $u \in \mathbb{F}_2^n$ such that $\bigoplus_{x \in X} x^u = 1$.

We reformulate the bit-based division property [Tod15b, TM16a] in terms of parity sets and the partial order framework.

Definition 5 (Bit-based division property)**.** A set $X \subseteq \mathbb{F}_2^n$ satisfies bit-based division property $\mathbb{K} \subseteq \mathbb{F}_2^n$, if

$$\text{ParitySet}\left(X\right) \subseteq \text{UpperClosure}\left(\mathbb{K}\right).$$

We define two special cases of division property mainly to simplify analysis.

Definition 6. For any set $X \subseteq \mathbb{F}_2^n$, define:

1. the *minimal division property* $\mathrm{MinDP}(X)$ of X as
$$\mathrm{MinDP}(X) := \mathrm{MinSet}(\mathrm{ParitySet}(X)),$$
2. the *full division property* $\mathrm{FullDP}(X)$ of X as
$$\mathrm{FullDP}(X) := \mathrm{UpperClosure}(\mathrm{ParitySet}(X)).$$

Boura and Canteaut developed distinguishers based on $\mathrm{UpperClosure}(\mathrm{ParitySet}(X))$, however the link with the bit-based division property was not explicitly established. In fact, they showed [BC16, Prop.6] that $\mathrm{UpperClosure}(\mathrm{ParitySet}(X))$ is precisely what is preserved when X goes through a constant addition:

$$\mathrm{UpperClosure}(\mathrm{ParitySet}(X \oplus c)) = \mathrm{UpperClosure}(\mathrm{ParitySet}(X))$$

for all $c \in \mathbb{F}_2^n$. It follows that bit-based division property is essentially equivalent to parity sets in the presence of key additions.

3.2 Link with the set indicator

We first note that the parity set of a set is closely linked to the ANF of the indicator of the set.

Proposition 3. *Parity set's coefficients can be expressed in terms of the ANF (Möbius) transform in the reverse direction:*

$$u \in \mathrm{ParitySet}(X) \quad \Leftrightarrow \quad \bigoplus_{x \succeq u} \mathbb{1}_X(x) = 1 \quad \Leftrightarrow \quad \bigoplus_{x \in \mathbb{F}_2^n} x^u \cdot \mathbb{1}_X(x) = 1.$$

Proof. The elements $x \in X$ contributing to the sum $\bigoplus_{x \in X} x^u = 1$ in Definition 4 are precisely those with $x \succeq u$. □

Corollary 1. *For any set $X \subseteq \mathbb{F}_2^n$,*

$$\mathrm{ParitySet}(X) = \neg\mathrm{Supp}_{\mathrm{ANF}}(\mathbb{1}_{\neg X}).$$

Several works [BKP16, GRW16, BC16] established independently the relation between the degree of a set and its state-level division property. Let D_k^n consist of all vectors of $\mathbb{F}_2^n$ of weight at least k. Then, a set $X \subseteq \mathbb{F}_2^n$ satisfies the division property D_k^n if and only if the degree of the indicator $\mathbb{1}_X$ of the set is at most $n-k$. The following proposition generalizes this relation to the case of bit-based division property. Naturally, *minimal vectors* of a bit-based division property define *maximal monomials* that can occur in the ANF of the indicator. As minimal/maximal vectors are compact representations of upper/lower sets, the same fact holds also for the respective closures.

Proposition 4. *Let $X \subseteq \mathbb{F}_2^n$. Then,*

$$\mathrm{MinDP}(X) := \mathrm{MinSet}(\mathrm{ParitySet}(X)) = \neg\mathrm{MaxSet}(\mathrm{Supp}_{\mathrm{ANF}}(\mathbb{1}_X)),$$
$$\mathrm{UpperClosure}(\mathrm{ParitySet}(X)) = \neg\mathrm{LowerClosure}(\mathrm{Supp}_{\mathrm{ANF}}(\mathbb{1}_X)).$$

Proof. Follows from Corollary 1, Proposition 2 and the fact that the set of maxterms in the ANF does not change on adding a constant to the input:

$$\begin{aligned}&\mathrm{MinSet}(\mathrm{ParitySet}(X)) = \mathrm{MinSet}(\neg\mathrm{Supp}_{\mathrm{ANF}}(\mathbb{1}_{\neg X}))\\ =&\neg\mathrm{MaxSet}(\mathrm{Supp}_{\mathrm{ANF}}(\mathbb{1}_{\neg X})) = \neg\mathrm{MaxSet}(\mathrm{Supp}_{\mathrm{ANF}}(\mathbb{1}_X)).\end{aligned}$$

□

More generally, an arbitrary division property $\mathbb{K}$ of a set X defines vectorial *upper bounds* on monomials in the indicator's ANF.

Corollary 2. *Let $X \subseteq \mathbb{F}_2^n$ such that X satisfies division property $\mathbb{K} \subseteq \mathbb{F}_2^n$. Then,*

$$\mathrm{Supp}_{\mathrm{ANF}}\left(\mathbb{1}_X\right) = \neg\mathrm{ParitySet}\left(\neg X\right) \subseteq \mathrm{LowerClosure}\left(\neg\mathbb{K}\right).$$

3.3 Division property propagation

Xiang *et al.* [XZBL16] proposed a method to propagate division property through a public function (an S-box). Essentially the same method was described by Boura and Canteaut in terms of parity sets, although not linked to the division property. We define division property transitions based on these methods.

Definition 7 (Division property transition)**.** Let $S : \mathbb{F}_2^n \to \mathbb{F}_2^m, u \in \mathbb{F}_2^n, v \in \mathbb{F}_2^m$. We say that (u, v) is a valid division property transition for S and write $u \xrightarrow{S} v$ if there exist $u' \succeq u$, $v' \preceq v$, such that $S^{v'}(x)$ contains the monomial $x^{u'}$. Otherwise, we write $u \not\xrightarrow{S} v$.

The defined kind of transition corresponds to full division property in the output and is useful for analysis. In practice, minimal (reduced) output division property is used as it reduces the search space of trail search algorithms.

Definition 8 (Minimal transition)**.** Let $u \xrightarrow{S} v$. If v is minimal such vector, then we say that $u \xrightarrow{S} v$ is a *minimal* transition and write $u \xrightarrow[\text{min.}]{S} v$.

Transitions allow to propagate division property through a public function. Due to monotonicity of division property, the propagation can be done by propagating each element of division property set $\mathbb{K}$ into a set of elements of output division property and taking a union over all such sets. This is a standard "propagation rule" in the division property literature, and was also formulated in terms of parity sets in [BC16, Prop.7].

Proposition 5. *Let $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$ and let $X \subseteq \mathbb{F}_2^n$ satisfy division property $\mathbb{K} \subseteq \mathbb{F}_2^n$. Then, the odd-multiplicity elements of $S(X)$ satisfy division property $\mathbb{K}'$, with*

$$\mathbb{K}' = \bigcup_{u \in \mathbb{K}} \left\{ v \in \mathbb{F}_2^m \mid u \xrightarrow{S} v \right\}.$$

Remark 2. It is sufficient to consider minimal transitions $u \xrightarrow[\text{min.}]{S} v$ instead of all $u \xrightarrow{S} v$, however, even in this case the resulting division property $\mathbb{K}'$ is not guaranteed to be minimal and has to be reduced if required so by search algorithms.

3.4 Core transitions and their characterizations

In this subsection, we describe the key component of our work: a new compact description of the set of division property transitions of a function. This new description is rather natural and turns out to be equivalent to the minimal division property of the graph of the function, or, alternatively, to the set of maximal monomials in the ANF of the graph indicator of the function.

First, we define a new subclass of transitions, called *core* transitions, which are *minimal* transitions with additional *maximality* restriction of the input division property vector. The idea is that, by Definition 7, a valid transition $u \xrightarrow{S} v$ induces valid transitions $u' \xrightarrow{S} v$ for all $u' \preceq u$. As a result, it is sufficient to store transitions with *maximal* u and *minimal* v. Indeed, any minimal transition $u \xrightarrow{S} v$ can be covered by some maximal u' such that $u' \xrightarrow{S} v$ is a core transition.

Definition 9 (Core transitions)**.** Let $u \xrightarrow{S} v$. If (u, v) is (maximal,minimal) such pair, then we say that $u \xrightarrow{S} v$ is a *core* transition and write $u \xrightarrow[\text{core}]{S} v$.

Remark 3. Todo and Morii [TM16b] proposed alternative compact structure of division property transitions. Their idea is to group input division property vectors by the output division sets they propagate to. However, the main usage of their compact structure was in an ad-hoc exhaustive trail search. It is not clear if SAT/MILP-based trail search can profit from such a structure. Our structure, on the contrary, lends itself naturally to compact CNF/DNF/MILP encodings (see Section 4).

We now show that core transitions have rich equivalent characterizations in terms of the ANFs of products of outputs bits, in terms of the ANF of the graph indicator and, finally, in terms of the (minimal) division property of the graph of the function.

Lemma 1. *Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$, $u \in \mathbb{F}_2^n$. Then,*

$$\bigoplus_{x \in \mathbb{F}_2^n} x^u f(x) = 1 \tag{1}$$

and u is minimal such vector if and only if the ANF of f contains maximal monomial $x^{\neg u}$.

Proof. Let X be the support of f. By Proposition 3, (1) holds if and only if $u \in \mathrm{ParitySet}(X)$. By Proposition 4, the vector u is minimal in $\mathrm{ParitySet}(X)$ if and only if $\neg u$ is maximal in $\mathrm{Supp}_{\mathrm{ANF}}(\mathbb{1}_X) = \mathrm{Supp}_{\mathrm{ANF}}(f)$. □

Theorem 1. *Let $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$, $u \in \mathbb{F}_2^n$, $v \in \mathbb{F}_2^m$. The following statements are equivalent:*

1. *$u \xrightarrow[\text{core}]{S} v$ (i.e., (u, v) is (maximal,minimal) such that $u \xrightarrow{S} v$);*
2. *(u, v) is (maximal,minimal) such that $S^v(x)$ contains the monomial x^u;*
3. *$(\neg u, v)$ belongs to $\mathrm{DivCore}_S := \mathrm{MinDP}(\mathbf{\Gamma}_S) := \mathrm{MinSet}(\mathrm{ParitySet}(\mathbf{\Gamma}_S))$; (Definition 10 below)*
4. *the graph indicator $\mathbb{1}_{\mathbf{\Gamma}_S}(x, y)$ contains the maximal monomial $x^u y^{\neg v}$.*

Proof. (1 ⇔ 2) Observe that $u \xrightarrow[\text{core}]{S} v$ implies that $S^v(x)$ contains the monomial x^u. Conversely, if $S^v(x)$ contains the monomial x^u, then $u \xrightarrow{S} v$. It follows that the extremality is transferred in both directions.

(2 ⇔ 3) By Definition 4, $(\neg u, v) \in \mathrm{MinSet}(\mathrm{ParitySet}(\mathbf{\Gamma}_S))$ if and only if

$$\bigoplus_{(x,y) \in \mathbf{\Gamma}_S} x^{\neg u} y^v = \bigoplus_{x \in \mathbb{F}_2^n} x^{\neg u} S^v(x) = 1 \tag{2}$$

and $(\neg u, v)$ is minimal such pair. For any fixed v, by Lemma 1, (2) holds with $\neg u$ minimal if and only if $S^v(x)$ contains the maximal monomial x^u. It follows that the extremality is transferred in both directions.

(3 ⇔ 4) Follows from Proposition 4 applied to the set $\mathbf{\Gamma}_S$. □

Remark 4. The extremality conditions are crucial and the proposed statements without them are not generally equivalent. On the other hand, the statements without extremality conditions hold instead for the respective upper/lower/mixed closures.

Remark 5. While characterizations 1 and 2 are related simply by definition, the other relations are more interesting. Remarkably, (1 ⇔ 3) identifies division property propagation through S with the (minimal) division property of the graph of S; (2 ⇔ 4) identifies *extreme* exponents (u, v) such that $S^v(x)$ contains x^u with maximal monomials in the graph indicator of S.

Note that the asymmetry of maximality/minimality of u/v is not present in characterizations 3 and 4: valid division property transitions of both S and S^{-1} (if it exists) are determined by the same set of *minimal* vectors $(\neg u, v) \in \mathrm{ParitySet}\,(\mathbf{\Gamma}_S)$, or, equivalently, by the same set of *maximal* monomials $x^u y^{\neg v}$ in the ANF of the graph indicator of S. This yields the following proposition.

Proposition 6. *Let S be a permutation of $\mathbb{F}_2^n$, $u, v \in \mathbb{F}_2^n$. Then,*

$$u \xrightarrow{S} v \quad \text{if and only if} \quad \neg v \xrightarrow{S^{-1}} \neg u,$$
$$u \xrightarrow[\text{core}]{S} v \quad \text{if and only if} \quad \neg v \xrightarrow[\text{core}]{S^{-1}} \neg u.$$

Proof. If $u \xrightarrow{S} v$, then by Definition 7 there exist $u' \succeq u, v' \preceq v$ such that $u' \xrightarrow[\text{core}]{S} v'$ and then by Theorem 1

$$\bigoplus_{(x,y)\in\mathbf{\Gamma}_S} x^{\neg u'} y^{v'} = 1.$$

By swapping roles of x, y, we obtain $\neg v' \xrightarrow{S^{-1}} \neg u'$. Since $\neg u' \preceq \neg u, \neg v' \succeq \neg v$, we get $\neg v \xrightarrow{S^{-1}} \neg u$. Equivalence for core transitions holds because the extremality condition is the same for both directions: $(\neg u, v)$ is minimal. □

Remark 6. This result is an extension of [BC16, Lemma 3] to the framework of division property transitions and extremality. The cited lemma states that $S^v(x)$ contains x^u if and only if $(\neg S^{-1})^{\neg u}(\neg x)$ contains $x^{\neg v}$. Furthermore, a similar degree-based statement was given by Boura and Canteaut already in [BC13].

Importantly, this proposition shows a bijection between forward and backward integral distinguishers based on division property. While this relation was known before, it is unfortunately rarely used in the literature to convert discovered forward distinguishers into backward distinguishers.

3.5 Division core and its relation to transition classes

From now on, we focus on studying the set of *core* transitions. Due to the aforementioned symmetry, it is more convenient to study its characterization as the the min-set of the parity set of the graph of S. As we shall use this set extensively, we introduce a new term for brevity.

Definition 10 (Division Core)**.** Let $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$. Define the *division core* of S, denoted $\mathrm{DivCore}_S$, as the minimal division property of the graph of S:

$$\mathrm{DivCore}_S \coloneqq \mathrm{MinDP}\,(\mathbf{\Gamma}_S) = \mathrm{MinSet}\,(\mathrm{ParitySet}\,(\mathbf{\Gamma}_S)) =$$
$$= \mathrm{MinSet}\left(\left\{(u, v) \in \mathbb{F}_2^n \times \mathbb{F}_2^m \;\middle|\; \bigoplus_{(x,y)\in\mathbf{\Gamma}_S} x^u y^v = 1\right\}\right).$$

We deduce the following characterization of division property transitions *solely from the division core.*

Theorem 2. *Let $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$. Then,*

1. $u \xrightarrow{S} v$ *if and only if* $(\neg u, v) \in \text{UpperClosure}\,(\text{DivCore}_S)$;
2. $u \xrightarrow[\text{min.}]{S} v$ *if and only if* $(\neg u, v) \in \text{MinSet}_v(\text{UpperClosure}\,(\text{DivCore}_S))$;
3. $u \xrightarrow[\text{core}]{S} v$ *if and only if* $(\neg u, v) \in \text{DivCore}_S$.

If, in addition, $n = m$ and S is bijective:

4. $v \xrightarrow{S^{-1}} u$ *if and only if* $(u, \neg v) \in \text{UpperClosure}\,(\text{DivCore}_S)$;
5. $v \xrightarrow[\text{min.}]{S^{-1}} u$ *if and only if* $(u, \neg v) \in \text{MinSet}_u(\text{UpperClosure}\,(\text{DivCore}_S))$;
6. $v \xrightarrow[\text{core}]{S^{-1}} u$ *if and only if* $(u, \neg v) \in \text{DivCore}_S$.

Here, the subscript of MinSet *defines the variable on which the min-set is computed (the vectors are labeled (u, v)).*

On the compactness of division core. By Sperner's theorem, the division core, as a min-set, has size bound $\mathcal{O}(2^{n+m}/\sqrt{n+m})$. This might seem as not so "compact" representation. For example, linear functions with domain $\mathbb{F}_2^n$ contain only vectors of weight n (to show this, consider any minimal transition $u \xrightarrow[\text{min.}]{S} v$ and observe that $\mathbf{wt}(\neg u) + \mathbf{wt}(v) = n$). Furthermore, for a random binary matrix $\mathbb{F}_2^{n \times m}$ one can expect a large number of invertible submatrices which translates into a large number of minimal/compact division property transitions (see [ZR18, HWW20]). Perhaps counter-intuitively, it follows that linear maps are the ones that may achieve the largest size of the division core, which could be interpreted as having the most complex division property propagation. On the opposite side, for a random function of full degree, most minimal transitions $u \xrightarrow[\text{min.}]{S} v$ have v of very small weight which translates into small-weight vectors in division core. This in turn makes most vectors of larger weight redundant and so the division core is expected to be a small set. The right intuition is that "heavier" functions tend to have "simpler" division property propagation and this is exactly captured by the division core as a compact representation.

Finally, we describe a new view on division trail composition in terms of the division core.

Proposition 7. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, $G : \mathbb{F}_2^m \to \mathbb{F}_2^r$, $u \in \mathbb{F}_2^n, w \in \mathbb{F}_2^r$. Then, there exists a valid division trail*

$$u \xrightarrow{F} v \xrightarrow{G} w$$

if and only if there exist $a \in \mathbb{F}_2^n, b, b' \in \mathbb{F}_2^m, c \in \mathbb{F}_2^r$ such that

$$a \preceq \neg u, \qquad (a, b) \in \text{DivCore}_F, \qquad b \wedge b' = \underline{0}, \qquad (b', c) \in \text{DivCore}_G, \qquad c \preceq w.$$

Proof. From $u \xrightarrow{F} v$ there must exist $u' \xrightarrow[\text{core}]{F} v'$ with $u' \succeq u, v' \preceq v$. Let $a = \neg u', b = v'$. Then, by Theorem 1, we have $(a, b) \in \text{DivCore}_F$. From $v \xrightarrow{G} w$ there must exist $v'' \xrightarrow[\text{core}]{G} w'$ with $v'' \succeq v, w' \preceq w$. Let $b' = \neg v'', c = w'$. Then, we have $(b', c) \in \text{DivCore}_G$. Since $v' \preceq v \preceq v''$, we have $b \wedge b' \coloneqq v' \wedge \neg v'' = \underline{0}$.

The other direction is analogous. The constraint $b \wedge b' = \underline{0}$ implies that $b \preceq \neg b'$ and then there exists a vector $v \in \mathbb{F}_2^m$ such that $b \preceq v \preceq \neg b'$, so that $(a, v) \in \text{UpperClosure}\,(\text{DivCore}_F)$, $(\neg v, c) \in \text{UpperClosure}\,(\text{DivCore}_G)$, implying the trail $\neg a \xrightarrow{F} v \xrightarrow{G} c$. Since $\neg a \succeq u$ and $c \preceq w$, the trail $u \xrightarrow{F} v \xrightarrow{G} w$ is also valid. □

3.6 Convex structure of the set of minimal transitions

In theory, identifying valid transitions (UpperClosure (DivCore$_S$)) is sufficient to identify propagation of division property and resulting integral distinguishers. In practice, it is crucial to also remove redundant transitions to reduce the search space of automated SAT/MILP solvers or ad-hoc search engines such as [Tod15b, TM16a, DF20]. Therefore, we analyze the set of *minimal/reduced* transitions in more details.

Definition 11. Let $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$. Define the following sets:

$$\begin{aligned}
\mathcal{I}_S &\coloneqq \left\{(u,v) \in \mathbb{F}_2^n \times \mathbb{F}_2^m \mid \neg u \overset{S}{\not\to} v\right\},\\
\mathcal{M}_S &\coloneqq \left\{(u,v) \in \mathbb{F}_2^n \times \mathbb{F}_2^m \mid \neg u \overset{S}{\to} v, \nexists v' \prec v : \neg u \overset{S}{\to} v'\right\},\\
\mathcal{R}_S &\coloneqq \left\{(u,v) \in \mathbb{F}_2^n \times \mathbb{F}_2^m \mid \neg u \overset{S}{\to} v, \exists v' \prec v : \neg u \overset{S}{\to} v'\right\}.
\end{aligned}$$

Remark 7. These sets contain respectively invalid transitions, minimal transitions and redundant transitions through S. The defining condition of $\mathcal{M}_S$ is equivalent to $\neg u \xrightarrow[\text{min.}]{S} v$.

Proposition 8. *The sets $\mathcal{I}_S, \mathcal{M}_S, \mathcal{R}_S$ form a partition of $\mathbb{F}_2^n \times \mathbb{F}_2^m$. Moreover, $\mathcal{I}_S$ is a lower set, $\mathcal{M}_S$ is a convex set, $\mathcal{R}_S$ is an upper set.*

Proof. The conditions of set generators in the sets' definitions clearly induce a partition of $\mathbb{F}_2^n \times \mathbb{F}_2^m$.

It is clear that $\mathcal{M}_S \cup \mathcal{R}_S = \text{UpperClosure}\,(\text{DivCore}_S)$ (both from the definitions and the fact that it is the complement of $\mathcal{I}_S$). Since $\mathcal{I}_S$ is the complement of this upper set, it must be a lower set.

The convexity of $\mathcal{M}_S$ follows from the fact that $\mathcal{M}_S = (\mathbb{F}_2^n \times \mathbb{F}_2^m) \setminus \mathcal{R}_S \setminus \mathcal{I}_S$. Indeed, let $a, c \in \mathcal{M}_S$. If there exists $b \notin \mathcal{M}_S$ such that $a \preceq b \preceq c$, then from $b \in \mathcal{I}_S$ it would follow that $a \in \mathcal{I}_S$ and so $a \notin \mathcal{M}_s$. The same argument applies to c and $\mathcal{R}_S$, leading to contradiction. □

We emphasize that all the three sets $\mathcal{I}_S, \mathcal{M}_S, \mathcal{R}_S$ can be derived from the division core DivCore$_S$, highlighting its universality as a compact representation:

$$\begin{aligned}
\mathcal{I}_S &= \overline{\text{UpperClosure}\,(\text{DivCore}_S)},\\
\mathcal{M}_S &= \text{MinSet}_v(\text{UpperClosure}\,(\text{DivCore}_S)),\\
\mathcal{R}_S &= \overline{\mathcal{I}_S \cup \mathcal{M}_S}.
\end{aligned}$$

Remarkably, these sets can themselves be expected to have compact representations in the form of max-set for $\mathcal{I}_S$, min-set for $\mathcal{R}_S$, and both min-set and max-set for $\mathcal{M}_S$. We discuss concrete efficient algorithms for computing these sets in Section 5.

Note that the maximal upper-set of removable vectors is given by

$$\mathcal{R}'_S \coloneqq \overline{\text{LowerClosure}\,(\mathcal{M}_S)}.$$

Compared to $\mathcal{R}_S$, it may include some extra vectors from $\mathcal{I}_S$ (but it always is a superset of $\mathcal{R}_S$). While its size is not smaller than that of $\mathcal{R}_S$, most often it has a simpler structure resulting in smaller models, as we shall see later on examples (see Table 1 in Section 4).

3.7 Linear combinations at the input/output

Lambin, Derbez and Fouque [LDF20] noticed that division property is not preserved under a composition of S-boxes with linear maps. One has to consider such maps in order to find integral distinguishers with a non-cube-shaped affine space at the input and/or a

balanced linear combination of bits at the output. The authors of [LDF20] exhausted all 4-bit linear maps to be composed with one S-box at the input and one S-box at the output. In [DF20], Derbez and Fouque showed that exhaustion of linear *maps* is unnecessary and exhaustion of linear *masks* is sufficient for finding maximal integral distinguishers, tremendously reducing the complexity.

For the input linear masks, they use the fact that an affine space of dimension $n-1$ can be defined by its 1-dimensional orthogonal complement, i.e. by its single non-zero vector. It is thus sufficient to define a linear bijective map that maps this vector to a single bit (completed arbitrarily), compose its inverse at the input of an S-box in the first round (and recompute the division property propagation through the composition), and assume this bit to be a constant and all other bits to be active in the division trail search.

For the output linear masks, the approach is more straightforward: define a bijective linear map that maps the chosen linear combination to a single bit, compose it at the output of an S-Box in the last round (and recompute the division property propagation through the composition), and, finally, check if this single output bit is balanced.

3.7.1 Formulaton in our framework

We now formulate this problem and simplify its solution in our framework. For simplicity, we assume that an "S-box" covers the full state. The case when target S-boxes cover only part of the state follows naturally. Our analysis is restricted to using traditional division property to find such distinguishers.

Let $S_{\text{in}} : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a bijection, $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, $S_{\text{out}} : \mathbb{F}_2^m \to \mathbb{F}_2^r$. Let $\alpha \in \mathbb{F}_2^n, \beta \in \mathbb{F}_2^r$ be the input and the output linear masks respectively, $\alpha \neq \underline{0}, \beta \neq \underline{0}$. We are interested in the integral (zero-sum) distinguishers of $S_{\text{out}} \circ F \circ S_{\text{in}}$ with the input linear mask α and the output mask β:

$$\bigoplus_{x \in \mathbb{F}_2^n, \langle \alpha, x \rangle = c} \langle \beta, S_{\text{out}} \circ F \circ S_{\text{in}}(x) \rangle = 0, \qquad c \in \mathbb{F}_2 \text{ a constant.}$$

The approach of [LDF20, DF20] is to search for division trails through each of the three steps of the composition

$$(\langle \beta, S_{\text{out}} \rangle) \;\circ\; F \;\circ\; (S_{\text{in}} \circ L_\alpha^{-1}),$$

where $L_\alpha \in GL_n(\mathbb{F}_2)$ is any such that the first coordinate of $L_\alpha(x)$ equals to $\langle \alpha, x \rangle$. To ensure the precision, the first and the last step must be propagated as units. The following theorem states an equivalent to the method of [DF20] sufficient condition of existence of such an integral distinguisher based on division property. As we will show in Subsection 4.3, this leads to easy and efficient CNF/MILP modeling.

Theorem 3. *Let $L_\alpha \in GL_n(\mathbb{F}_2)$ be such that $L_\alpha(x) = (\langle \alpha, x \rangle, \dots)$ Then, there exists a division trail*

$$(0, 1, \dots, 1) \xrightarrow{S_{in} \circ L_\alpha^{-1}} u \xrightarrow{F} v \xrightarrow{\langle \beta, S_{out} \rangle} (1)$$

if and only if $u \xrightarrow{F} v$ and

$$\neg u \in \text{LowerClosure}\left(\text{Supp}_{\text{ANF}}\left\langle \alpha, S_{in}^{-1} \right\rangle\right), \tag{3}$$

$$v \in \text{LowerClosure}\left(\text{Supp}_{\text{ANF}} \langle \beta, S_{out} \rangle\right). \tag{4}$$

Proof. The first transition by Proposition 6 is equivalent to $\neg u \xrightarrow{L_\alpha \circ S_{\text{in}}^{-1}} (1, 0, \dots, 0)$, equivalently $\neg u \xrightarrow{\left\langle \alpha, S_{\text{in}}^{-1} \right\rangle} (1)$, equivalent to (3). The last transition is similarly equivalent to (4). □

Remark 8. For a non-invertible $S_{\text{in}} : \mathbb{F}_2^{n'} \to \mathbb{F}_2^n$, one can replace the Boolean function $y \mapsto \langle \alpha, S_{\text{in}}^{-1}(y) \rangle$ by the function

$$y \mapsto \bigoplus_{x \in (S_{\text{in}} \circ L_\alpha^{-1})^{-1}(y)} \langle \alpha, x \rangle .$$

3.8 Relationships with graph indicator-based degree bounds

Recently, Carlet [Car20a] derived new degree bounds on compositions of functions based on the degrees of the graph indicators of the involved functions. It is a natural question whether these bounds can beat traditional bit-based division property and whether division property can be improved by incorporating these bounds. In this section, we show a close relationship of these bounds with division property propagations, based on the relationship of division property propagation and the graph indicator given by Theorem 1.

Carlet in [Car20b] gives an elegant expression for the graph indicator of the composition of functions in terms of their graph indicators.

Proposition 9 ([Car20b, Car20a]). *Let $G_i : \mathbb{F}_2^{m_{i-1}} \to \mathbb{F}_2^{m_i}$, $i \in \{1, \dots, r\}$, let $F = G_r \circ \dots \circ G_1$. Then,*

$$\mathbb{1}_{\mathbf{\Gamma}_F}(x, z) = \bigoplus_{\substack{(y_1, \dots, y_{r-1}) \\ \in \mathbb{F}_2^{m_1} \times \dots \times \mathbb{F}_2^{m_{r-1}}}} \mathbb{1}_{\mathbf{\Gamma}_{G_1}}(x, y_1) \cdot \mathbb{1}_{\mathbf{\Gamma}_{G_2}}(y_1, y_2) \cdot \dots \cdot \mathbb{1}_{\mathbf{\Gamma}_{G_r}}(y_{r-1}, z).$$

Example 3. Let $H : \mathbb{F}_2^n \to \mathbb{F}_2^m, G : \mathbb{F}_2^m \to \mathbb{F}_2^r$. Then,

$$\mathbb{1}_{\mathbf{\Gamma}_{G \circ H}}(x, z) = \bigoplus_{y \in \mathbb{F}_2^m} \mathbb{1}_{\mathbf{\Gamma}_H}(x, y) \mathbb{1}_{\mathbf{\Gamma}_G}(y, z).$$

This expression naturally allows to bound possible monomials in $\mathbb{1}_{\mathbf{\Gamma}_F}(x, z)$: (i) $\mathbb{1}_{\mathbf{\Gamma}_F}(x, z)$ does not contain a monomial multiple of $x^u z^{\neg v}$ if and only if (ii)

$$\mathbb{1}_{\mathbf{\Gamma}_{G_1}}(x, y_1) \mathbb{1}_{\mathbf{\Gamma}_{G_2}}(y_1, y_2) \dots \mathbb{1}_{\mathbf{\Gamma}_{G_r}}(y_{r-1}, z)$$

does not contain a monomial multiple of $x^u y_1^{m_1} y_2^{m_2} \dots y_{r-1}^{m_{r-1}} z^{\neg v}$. By Theorem 1, the condition (i) is equivalent to: for any $v' \preceq v$, $F^{v'}(x)$ does not contain a monomial multiple of x^u. Sufficient conditions for (ii) can be derived from degree bounds of the involved graph indicators, as done in [Car20a]. In this way, graph indicators' degrees allow to derive upper bounds on monomials occurring in products of outputs of the composition F.

We now show that bit-based division property verifies a stronger condition, which in fact can be seen as a bit-based formulation of the degree-based bounds.

Theorem 4. *Let F, G_i be defined as above. Let I be the formal expansion (i.e., no $\oplus$-cancellations) of*

$$\mathbb{1}_{\mathbf{\Gamma}_{G_1}}(x, y_1) \mathbb{1}_{\mathbf{\Gamma}_{G_2}}(y_1, y_2) \dots \mathbb{1}_{\mathbf{\Gamma}_{G_r}}(y_{r-1}, z).$$

Then, I contains a monomial multiple of

$$x^u y_1^{m_1} y_2^{m_2} \dots y_{r-1}^{m_{r-1}} z^{\neg v} \tag{5}$$

if and only if there exists a valid division trail

$$u \xrightarrow{G_1} w_1 \xrightarrow{G_2} \dots \xrightarrow{G_{r-1}} w_{r-1} \xrightarrow{G_r} v. \tag{6}$$

Proof. By Theorem 1, each link in the trail has an equivalent condition on the monomial multiple in the corresponding graph indicator:

$$\begin{aligned} u \xrightarrow{G_1} w_1 &\quad\Leftrightarrow\quad \mathbb{1}_{\mathbf{\Gamma}_{G_1}}(x, y_1) \text{ contains a monomial multiple of } x^u y_1^{\neg w_1}, \\ w_1 \xrightarrow{G_2} w_2 &\quad\Leftrightarrow\quad \mathbb{1}_{\mathbf{\Gamma}_{G_2}}(y_1, y_2) \text{ contains a monomial multiple of } y_1^{w_1} y_2^{\neg w_2}, \\ &\dots \\ w_{r-1} \xrightarrow{G_r} v &\quad\Leftrightarrow\quad \mathbb{1}_{\mathbf{\Gamma}_{G_{r-1}}}(y_{r-1}, z) \text{ contains a monomial multiple of } y_{r-1}^{w_{r-1}} z^{\neg v}. \end{aligned}$$

($\Rightarrow$) If I contains a monomial multiple of (5), there exists one monomial per each of $\mathbb{1}_{\mathbf{\Gamma}_{G_1}}, \mathbb{1}_{\mathbf{\Gamma}_{G_2}}, \dots$ such that all these monomials multiply to (5). Clearly, there must exist $w_1, \dots, w_{r-1}$ such that $\mathbb{1}_{\mathbf{\Gamma}_{G_1}}(x, y_1)$ contains a monomial multiple of $x^u y_1^{\neg w_1}$, $\mathbb{1}_{\mathbf{\Gamma}_{G_2}}(y_1, y_2)$ contains a monomial multiple of $y_1^{w_1} y_2^{\neg w_2}$ (to get $y_1^{m_1}$), etc.

($\Leftarrow$) If there exists a trail of the form (6), then there exist corresponding monomial multiples of $x^u y_1^{\neg w_1}$, $y_1^{w_1} y_2^{\neg w_2}$, etc. that obviously multiply to a monomial multiple of (5). □

This theorem gives an alternative view on division property trails: a division property trail $u \xrightarrow{G_1} \dots \xrightarrow{G_r} v$ is equivalent to a chain of monomials, one from each of the graph indicators of the composed functions $G_1, \dots, G_r$, such that, in their product, all intermediate variables are fully saturated, the input variable has an exponent succeeding u and the output variable has an exponent succeeding $\neg v$. In particular, division property allows to derive an upper bound on monomials occurring in the graph indicator of the composition.

While an existence of such a trail / a monomial chain does not mean that $\mathbb{1}_{\mathbf{\Gamma}_F}$ in fact *contains* a monomial multiple of $x^u y^{\neg v}$ (due to the possible cancellations), the inverse is true: for $\mathbb{1}_{\mathbf{\Gamma}_F}$ to contain such a monomial multiple, there must exist a corresponding division trail.

We conclude that traditional bit-based division property is optimal in determining upper bounds on monomials in $\mathbb{1}_{\mathbf{\Gamma}_F}$ *as long as cancellations in the product*

$$\mathbb{1}_{\mathbf{\Gamma}_{G_1}}(x, y_1)\mathbb{1}_{\mathbf{\Gamma}_{G_2}}(y_1, y_2) \dots \mathbb{1}_{\mathbf{\Gamma}_{G_r}}(y_{r-1}, z)$$

are not considered.

4 CNF modeling of a convex set

In this section, we show that the convex structure of division property transitions from Subsection 3.6 naturally lends itself to CNF models. We recall that it is sufficient to derive constraints removing the lower set $\mathcal{I}_S$ and the upper set $\mathcal{R}_S$ (or $\mathcal{R}'_S$).

Remark 9. Any CNF formula can be trivially converted to a MILP system, however MILP inequalities are generally more expressive and one can expect a significant reduction in the number of inequalities compared to the number of clauses. Recently, Udovenko [Udo21] developed techniques for constructing smallest MILP models for Boolean functions. In particular, an efficient approach for modeling monotone Boolean functions (lower/upper sets) is given and can be directly applied to remove the lower set $\mathcal{I}_S$ and the upper set $\mathcal{R}_S/\mathcal{R}'_S$ optimally (separately).

Throughout this section, we consider division property transitions in the "directionless" (symmetric) way: for a transition $u \xrightarrow{S} v$, we consider the vector $(\neg u, v)$. This is done for convenience and has no extra cost since the variable negation is free in CNF/MILP models.

4.1 Basic modeling

A lower set W is called *principal* if it is spanned by a single element: $W = \text{LowerClosure}(\{w\})$. Such a lower set can be removed by one CNF clause precisely without removing any other point from the hypercube $\{0,1\}^n$. In fact, up to negation of the variables, a principal lower set is exactly what can be removed by a single CNF clause. It is thus a building block of general CNF modeling tools such as the Quine-McCluskey algorithm [Qui55, McC56].

Proposition 10. *Let $w \in \mathbb{F}_2^n$. Then,*

$$x \notin \text{LowerClosure}(\{w\}) \iff \bigvee_{i:w_i=0} x_i$$
$$x \notin \text{UpperClosure}(\{w\}) \iff \bigvee_{i:w_i=1} \neg x_i.$$

Since a general lower set is a union of principal lower sets by definition, it can be removed by a set of clauses each removing a principal lower set spanned by one of the maximal elements. The case of an upper set is completely analogous.

Corollary 3. *The set $\mathcal{M}_S$ of minimal division property transitions can be modeled by $|\text{MaxSet}(\mathcal{I}_S)| + |\text{MinSet}(\mathcal{R}_S)|$ constraints (CNF clauses or integer inequalities).*

It is also easy to show that such CNF model is *optimal* (in the number of clauses), although *separately* for each of the two sets $\mathcal{I}_S$ and $\mathcal{R}_S$.

Proposition 11. *Let $L \subseteq \mathbb{F}_2^n$ be a lower set. If a CNF formula precisely removes L from the hypercube $\{0,1\}^n$, then it contains at least $|\text{MaxSet}(L)|$ clauses.*

Proof. Let $a \oplus \text{LowerClosure}(w), a \wedge w = \underline{0}$ be the cube removed by a clause in the formula modeling L. Observe that

$$a \oplus \text{LowerClosure}(w) \subseteq \text{LowerClosure}(a \vee w) \subseteq L,$$

where the second inclusion follows from the monotonicity of L. Therefore, we can replace all clauses in the formula by monotone ones (i.e. with $a = \underline{0}$). Observe that each such *principal* lower set can remove at most one element from $\text{MaxSet}(L)$ (without removing anything from $\overline{\text{LowerClosure}(L)}$). The proposition follows. □

We provide the sizes of the relevant sets for a variety of S-boxes in Table 1. For optimal CNF encodings, we used the Quine-McCluskey algorithm together with the open source SCIP optimization suite [GAB+20] to find/bound the minimum number of clauses (approach described in [BC20]).

Example 4. Consider the AES S-Box $S : \mathbb{F}_2^8 \to \mathbb{F}_2^8$ as an example. Its division core DivCore_S contains 122 vectors $(u, v) \in \mathbb{F}_2^8 \times \mathbb{F}_2^8$ with $(\mathbf{wt}(u), \mathbf{wt}(v))$ distributed as follows:

$$\begin{array}{llll} (0,8):1, & (1,1):25, & (1,2):40, & (1,3):6, \\ (2,1):26, & (2,2):4, & (3,1):19, & (8,0):1. \end{array}$$

Here, weights $(8,0)$ and $(0,8)$ correspond to the vectors $(\underline{1},\underline{0})$, $(\underline{0},\underline{1})$ which in turn correspond to the division property of the domain and of its image. The set $\text{MaxSet}(\mathcal{I}_S)$ contains 87 maximal invalid vectors, the $\text{MinSet}(\mathcal{R}_S)$ contains 319 minimal redundant vectors. Therefore, minimal transitions through S can be precisely described by 406 CNF clauses (and 87 are sufficient at the cost of allowing redundant transitions). Using the alternative upper bound $\text{MinSet}(\mathcal{R}'_S)$ allows to further reduce the number to 87+274=361 clauses.

We compare briefly with other tools/methods. The automated tool Solvatore [EKKT19] generates 2921 CNF clauses. A tool from Hu-Wang-Wang [HWW20] uses the STP solver and generates a DNF formula by enumerating all 2001 valid non-redundant trails. Our approach can be easily adapted to compute two DNF formulas with much less clauses: $122 + 119 = 241$. With the Quine-McCluskey algorithm (applied to division property in [GD21]) we obtain the optimal value of 234 CNF and a heuristic value of $\leq$151 DNF clauses. This is about 2 times better than our result, showing however that our models are close to optimal (in particular, removing invalid and redundant trails separately is done optimally by Proposition 11). Most importantly, Quine-McCluskey is not applicable to larger S-boxes while our method can produce CNF/DNF models of very good quality.

Table 1: Sizes of the convex sets relevant for modeling division property for a variety of S-boxes. MinDPPT_S is the set of all minimal division property transitions. DivCore_S is the compact set containing all the information about division property transitions. $\text{MaxSet}(\mathcal{I}_S)$ and one of $\text{MinSet}(\mathcal{R}_S)$, $\text{MinSet}(\mathcal{R}'_S)$ define the number of CNF clauses sufficient for SAT modeling (see Section 4). † since MixColumn of Midori-64/Skinny-64 consist of 4 parallel independent 4-bit maps, the optimal CNF was computed from concatenating 4 optimal CNF models (28/21 clauses respectively) of each 4-bit block.

func. S	n	$\lvert\text{MinDPPT}_S\rvert$	$\lvert\text{DivCore}_S\rvert$	$\lvert\text{MaxSet}(\mathcal{I}_S)\rvert$	$\lvert\text{MinSet}(\mathcal{R}_S)\rvert$	$\lvert\text{MinSet}(\mathcal{R}'_S)\rvert$	CNF (our)	CNF (opt.)
Present	4	47	16	20	24	24	44	26
Knot	4	49	26	32	29	27	59	40
Ascon	5	190	71	83	93	83	166	115
Keccak	5	137	57	45	75	25	70	50
Fides	6	419	188	146	359	254	400	222
Misty S7	7	1779	436	396	1000	967	1363	607
AES	8	2001	122	87	319	274	361	234
Skinny-128	8	2089	611	193	1383	198	391	246
DryGASCON-256	9	7983	631	480	1309	552	1032	475
Misty S9	9	27 623	6755	5120	18 575	16 868	21 988	10403-11819
LED MixColumn	16	177 643 913	177 643 913	33 412	334 974 429	33 061	66 473	-
Midori-64 MixColumn	16	9 834 496	9 834 496	56	39 337 984	56	112	112†
Skinny-64 MixColumn	16	1 185 921	1 185 921	40	6 324 912	44	84	84†
Midori-64 Super-Sbox (all keys)	16	14 714 723	2 380 924	1 912 088	6 277 211	4 317 883	6 229 971	-
LED Super-Sbox (all keys)	16	8 458 909	319 606	321 168	1 119 494	1 261 465	1 440 662	-
LED Super-Sbox (zero key)	16	8 481 417	382 591	388 134	1 215 435	1 317 330	1 603 569	-

4.2 Cardinality bounds

Cardinality bounds allow to bound the number of bits equal to 1 among a given set of variables. A popular CNF construction for encoding cardinality bounds is due to Sinz [Sin05] and is based on the so-called sequential counters, which encode addition of integer variables in the unary representation. Although it requires auxiliary variables, it is known to perform well on practice, since it is decided by unit propagation. Cardinality bounds using sequential counters were used recently for differential/linear trail search using SAT-solvers [SWW21].

Cardinality bounds may be particularly helpful for constraining division property transitions, as they can remove a large number of transitions at a very low cost. There are two particular use cases.

The first use is to replace a precise convex upper bound (e.g., $\text{MinSet}(\mathcal{R}_S)$ or $\text{MinSet}(\tilde{\mathcal{R}}_S)$) by a simpler (yet possibly imprecise) cardinality upper bound. Here, we use the fact

that removing precisely *all* redundant transitions is not necessary: it is usually done as a heuristic aid for SAT solvers to reduce the search space. For a function S, this cardinality constraint is given by $\mathbf{wt}(u||v) \leq h$, where $h \coloneqq \max_{w \in \mathcal{M}_S} \mathbf{wt}(w)$ and u, v are the division property variables modeling the transition $\neg u \xrightarrow{S} v$.

The second use is to supplement precise bounds to allow faster conflicts during the SAT search. Cardinality bounds allow solvers to quickly skip a large part of invalid transitions, and to process the remaining precise constraints on the remaining smaller search space. In addition to the upper bound described above, a supplementary lower bound is given by $l \leq \mathbf{wt}(u||v)$, where $l \coloneqq \min_{w \in \mathcal{M}_S} \mathbf{wt}(w)$.

4.2.1 The case of a linear map

We consider the particular case of a linear map $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$. For a minimal transition $\neg u \xrightarrow[\text{min.}]{S} v$ it is known that $\mathbf{wt}(\neg u) = \mathbf{wt}(v)$ is necessary but not sufficient. In the symmetric form (u, v), this constraint becomes

$$n - \mathbf{wt}(u) = \mathbf{wt}(v) \quad \Leftrightarrow \quad \mathbf{wt}(u||v) = n.$$

A redundant transition (u, v) is such that $\mathbf{wt}(v) > \mathbf{wt}(\neg u)$, implying

$$\mathbf{wt}(u||v) > n.$$

It follows that redundant transitions $\mathcal{R}_S$ can be removed with a single cardinality constraint $\mathbf{wt}(u||v) \leq n$.

Proposition 12. *For a linear map $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$, for some $I \subseteq \mathcal{I}_S$, the set $\mathcal{R}_S \cup I$ can be removed with a single cardinality constraint $\mathbf{wt}(u||v) \leq n$, where $(u||v) \in \mathbb{F}_2^{2n}$.*

Remark 10. It is natural to use the more strict constraint $\mathbf{wt}(u||v) = n$, since it may also remove a larger part of $\mathcal{I}_S$.

Remark 11. This constraint is equivalent to $\mathbf{wt}(\neg u) = \mathbf{wt}(v)$ (for the transition $\neg u \xrightarrow{S} v$) and is basic and well-known in the literature. What is important for our purposes is that it fully removes $\mathcal{R}_S$.

Example 5. Consider the MixColumns matrix of LED [GPPR11], $M : \mathbb{F}_2^{16} \to \mathbb{F}_2^{16}$ (see Table 1). It is such that:

$$|\mathcal{M}_M| = 177\,643\,913; \qquad |\text{MinSet}\,(\mathcal{R}_M)| = 334\,974\,429;$$
$$|\text{MaxSet}\,(\mathcal{I}_M)| = 33\,412; \qquad |\text{MinSet}\,(\mathcal{R}'_M)| = 33\,061.$$

Despite a large number of minimal division property transitions (177M), it can be modeled by only 33k CNF clauses plus a cardinality constraint, which adds a negligible amount of clauses and auxiliary variables.

Remark 12. The approach of [HWW20] (using auxiliary variables) allows to model large linear layers (up to 64 bits), by encoding the submatrix invertability condition in the problem, in a way that requires the SMT solver to find the inverse matrix. We remark though that it was only presented in the SMT form, not in pure SAT or MILP.

The advantage of our SAT encoding (which although has a smaller feasible range of about 16-bit linear maps) is its simpler form and the fact that it can be decided by unit propagation: given the input and output mask (u, v), the SAT solver can decide its validity without making further guesses (although at the cost of verifying a possibly large number of clauses).

Multidimensional cardinality bounds Cardinality bounds may be generalized to multiple dimensions by bounding vectors of cardinalities of smaller chunks of the vector $(u||v)$. However, we did not notice a significant performance improvement of this method in our application to LED (see Section 6). The technique is described in Appendix C.

4.3 Linear masks at the input / at the output

In Subsection 3.7, we derived simple conditions for applying linear masks at the input and/or at the output. We now show how to model these conditions. Recall that we consider a composition $S_{\text{out}} \circ F \circ S_{\text{in}}$ with an input linear mask α and an output linear mask β. Theorem 3 provides the following necessary and sufficient conditions (together with the validity of $u \xrightarrow{F} v$):

$$\begin{aligned} \neg u &\in \text{LowerClosure}\left(\text{Supp}_{\text{ANF}}\left\langle \alpha, S_{\text{in}}^{-1} \right\rangle\right), \\ v &\in \text{LowerClosure}\left(\text{Supp}_{\text{ANF}}\left\langle \beta, S_{\text{out}} \right\rangle\right). \end{aligned}$$

These three conditions can be efficiently modeled by CNF/MILP formulas as was described in Subsection 4.1.

Moreover, it is sufficient to check if a transition $u \xrightarrow{F} v$ is valid for any of *maximal* exponents $\neg u, v$ in the ANFs of $\left\langle \alpha, S_{\text{in}}^{-1} \right\rangle$ and $\left\langle \beta, S_{\text{out}} \right\rangle$ respectively. However, the maximality of v can not be guaranteed in practice since the corresponding trail $u \xrightarrow{F} v$ may be redundant, while standard modeling approaches disallow redundant transitions for efficiency reasons.

For the input case, we can restrict the division property mask of the input to F to take values only from $\neg\text{MaxSet}\left(\text{Supp}_{\text{ANF}}\left\langle \alpha, S_{\text{in}}^{-1} \right\rangle\right)$, with the goal of reducing the search space. Since a max-set is an antichain, it is convex, and can be modeled by removing the complementary lower and upper bounds. Formally, define

$$\begin{aligned} U &\coloneqq \text{MaxSet}\left(\text{Supp}_{\text{ANF}}\left\langle \alpha, S_{\text{in}}^{-1} \right\rangle\right), \\ P &\coloneqq \text{MaxSet}\left(\overline{\text{UpperClosure}\left(U\right)}\right), \\ Q &\coloneqq \text{MinSet}\left(\overline{\text{LowerClosure}\left(U\right)}\right). \end{aligned}$$

Then, a vector $x \in \mathbb{F}_2^n$ belongs to U (we set $x \coloneqq \neg u$) if and only if

$$(x \notin \text{LowerClosure}\left(P\right)) \wedge (x \notin \text{UpperClosure}\left(Q\right)),$$

which can be encoded by $|P| + |Q|$ CNF clauses (or MILP inequalities).

5 Algorithmic framework for dense sets

5.1 Bitwise transformations, lower, upper, min-, max-sets

We start by introducing a simple yet very generic and powerful tool for manipulating dense subsets of $\mathbb{F}_2^n$ represented by their indicator vectors. This is a straightforward abstraction of well-known algorithms such as the Möbius transform for computing the ANF, the Walsh-Hadamard transform, sum-over-subsets technique, etc. The tool is described in Algorithm 1.

Algorithm 1 Bitwise multidimensional transform

Input: array $X \in A^{2^n}$, transformation map $f : A^2 \to A^2$, mask $I \in \mathbb{F}_2^n$ set to $\underline{1}$ by default
Output: in-place transformed array $X \in A^{2^n}$
Complexity: $\mathcal{O}(\mathbf{wt}(I)2^n) \leq \mathcal{O}(n2^n)$

```
function Transform[f, I](X)
    for all i ∈ {0, ..., n − 1}, s.t. I has i-th bit set do        ▷ 0-based
        for all j ∈ {0, ..., 2^n − 1}, s.t. j has (n − 1 − i)-th bit set do        ▷ 0-based
            (X_{j−2^i}, X_j) ← f(X_{j−2^i}, X_j)
    return X
```

Definition 12. Define the following maps with the signature $(\mathbb{F}_2)^2 \to (\mathbb{F}_2)^2$:

$$\begin{aligned}
\text{XOR-up} &: (a, b) \mapsto (a, b \oplus a),\\
\text{XOR-down} &: (a, b) \mapsto (a \oplus b, b),\\
\text{OR-up} &: (a, b) \mapsto (a, b \vee a),\\
\text{OR-down} &: (a, b) \mapsto (a \vee b, a),\\
\text{LESS-up} &: (a, b) \mapsto (a, b \wedge \neg a), \quad \text{equiv. } b \leftarrow b \wedge [a < b],\\
\text{MORE-down} &: (a, b) \mapsto (a \wedge \neg b, b), \quad \text{equiv. } a \leftarrow a \wedge [a > b].
\end{aligned}$$

Proposition 13. *The defined transformations have the following effects:*

1. Transform[XOR-up] *computes the Möbius transform (involution), i.e. transforms the truth table of a Boolean function into its ANF and vice versa.*
2. Transform[XOR-down] *computes the involution* ParitySet*;*
3. Transform[OR-up] *computes* UpperClosure*.*
4. Transform[OR-down] *computes* LowerClosure*.*
5. Transform[LESS-up] $\circ$ Transform[OR-up] *computes* MinSet*.*
6. Transform[MORE-down] $\circ$ Transform[OR-down] *computes* MaxSet*.*

Proof. The proofs can be done by induction on the bit-position. □

Remark 13. The transformations can be efficiently batched in an efficient bitslice fashion, by lifting the set A and operations from $\mathbb{F}_2$ to $\mathbb{F}_2^t$ where t is the number of considered sets.

5.2 Division property of a set

Malviya and Tiwari [MT21] consider the problem of computing the minimal division property of a given (multi)set X. They claim classical complexity $\mathcal{O}(n2^n|X|)$ and quantum complexity $\mathcal{O}(n2^n\sqrt{|X|})$.

The relation between the division property and the set indicator given by Proposition 4 together with the fast MinSet algorithm from the previous subsection lead to a simple and efficient classical algorithm with complexity $\mathcal{O}(n2^n)$ for the problem (see Algorithm 2).

5.3 Division core and propagation table

Let $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$. By definition, $\text{DivCore}_S \coloneqq \text{MinDP}(\mathbf{\Gamma}_S)$, which can be computed by Algorithm 2. This approach leads to time and memory complexity $\mathcal{O}((n+m)2^{n+m})$. In particular, for bijective S-Boxes we get the time complexity $\mathcal{O}(n2^{2n})$. The complexity is independent of the S-box and of the size of the division core.

Algorithm 2 Minimal division property of a set

Input: $X \subseteq \mathbb{F}_2^n$
Output: $\mathrm{MinDP}(X) \subseteq \mathbb{F}_2^n$
Complexity: $\mathcal{O}(n2^n)$

1: $G \leftarrow$ indicator vector of X $(\in \mathbb{F}_2^{2^n})$
2: $G \leftarrow \mathrm{Transform}[\text{XOR-down}](G)$ ▷ parity set of X
3: $G \leftarrow \mathrm{Transform}[\text{OR-up}](G)$ ▷ upper set of parity masks
4: $G \leftarrow \mathrm{Transform}[\text{LESS-up}](G)$ ▷ min-set of parity masks
5: **return** G ▷ $\mathrm{MinDP}(X)$

Recall that the set of all valid division property transitions through S can be computed as $(\underline{1}, \underline{0}) \oplus \mathrm{UpperClosure}(\mathrm{DivCore}_S)$. To obtain the usual reduced division property propagation table (i.e., all minimal transitions), we can simply compute partial min-set on the second coordinate. See Algorithm 3 for details.

Algorithm 3 Division property propagation table (only minimal transitions)

Input: $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$ as a lookup-table
Output: reduced DPPT of S: $D = \left\{(u, v) \in \mathbb{F}_2^n \times \mathbb{F}_2^m \mid u \xrightarrow[\text{min.}]{S} v\right\}$
Complexity: $\mathcal{O}((n+m)2^{n+m})$

1: $D \leftarrow$ indicator vector of $\mathbf{\Gamma}_S$ $(\in \mathbb{F}_2^{2^{n+m}})$
2: $D \leftarrow \mathrm{Transform}[\text{XOR-down}](D)$ ▷ $\mathrm{ParitySet}(\mathbf{\Gamma}_S)$
3: $D \leftarrow \mathrm{Transform}[\text{OR-up}](D)$ ▷ full DPPT (up to $\neg u$)
4: $D \leftarrow \mathrm{Transform}[\text{LESS-up}, (\underline{0}, \underline{1})](D)$ ▷ min-set on v; $= \mathcal{M}_S$ from Definition 11
5: **return** $D \leftarrow (\underline{1}, \underline{0}) \oplus D$ ▷ compute $\neg u$

This in particular achieves "quadratic" complexity $\mathcal{O}(n2^{2n})$, an improvement over the "cubic" complexity $\mathcal{O}(2^{3n})$ claimed in [DF20] for computing the DPPT using algorithm from [XZBL16] (in the case $m = n$).

Finally, from the set $\mathcal{M}_S$ computed by *Algorithm 3* we can easily compute the necessary min-/max-sets and respective complementary sets required for modeling:

$$\mathcal{I}_S = \overline{\mathrm{UpperClosure}(\mathrm{DivCore}_S)},$$
$$\mathcal{R}_S = \overline{\mathcal{I}_S \cup \mathcal{M}_S},$$
$$\mathcal{R}'_S = \overline{\mathrm{LowerClosure}(\mathrm{DivCore}_S)}.$$

For the compact CNF modeling (Section 4), it is left to compute $\mathrm{MaxSet}(\mathcal{I}_S)$ and $\mathrm{MinSet}(\mathcal{R}_S)$ (or $\mathrm{MinSet}(\mathcal{R}'_S)$).

5.4 Compact representation (advanced algorithm)

In this subsection, we describe a breadth-first search algorithm which performs much better for "heavy" functions, i.e., those having many high-degree monomials in most products of output bits, implying a small size of the division core and a small number of non-trivial invalid transitions. In this algorithm, we assume access to the lookup table of the function and the memory footprint is of the same magnitude, so this approach is limited up to about 32-bit functions on practice.

We restrict the description to the case of a bijective function $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ for simplicity, as non-bijective functions would require more fine-grained case analysis due to possible degeneracy.

We consider first vectors $(u, v) \in \mathrm{DivCore}_S$ with $u = \underline{0}$ or $v = \underline{0}$. The case of $v = \underline{0}$ corresponds to the minimal division property of the domain which leads exactly to $(\underline{1}, \underline{0}) \in \mathrm{DivCore}_S$. The case of $u = \underline{0}$ can be exhausted by computing the minimal division property of the image of S (more precisely, of the set of its elements with odd multiplicity). For bijective S this case leads to only $(\underline{0}, \underline{1}) \in \mathrm{DivCore}_S$. Note that all strict predecessors of these vectors define invalid transitions (have parity zero), and should be explicitly excluded to avoid enumeration of the $2 \cdot 2^n$ "trivial" pairs.

We are going to explore all possible nonzero u, v in a breadth-first manner (from low weight to high weight), until we obtain the full division core of S. Given a pair (u, v) of unknown parity, and a promise that all its strictly preceeding vectors have parity zero (due to the exploration order), we can compute its parity by computing the parity set of (the support of) S^v or of $(S^{-1})^u$; we choose the one with the minimal weight ($\mathbf{wt}(v)$ or $\mathbf{wt}(u)$). The parity set of, say, S^v, may provide many other vectors $(u', v) \in \mathrm{DivCore}_S$. In particular, we consider all *minimal* u' in the parity set as candidates and save the corresponding pairs (u', v) in a set D. Although D may also include redundant vectors, each vector of $\mathrm{DivCore}_S$ will be present in one of such lists of candidates.

After the main step, if (u, v) has parity one, we add it to the division core (it is guaranteed to be minimal due to the exploration order) and continue with the next pair in the queue. Otherwise, if (u, v) has parity zero, we consider its successors for adding to the exploration queue. However, for each pair, we maintain a counter of its direct predecessors that were visited and have parity zero. The pair is added to the queue only when the counter is full, i.e. when the last direct predecessor is visited. This allows to avoid duplicate processing of (u, v), and, more importantly, ensures that all predecessors have parity zero and the new pair is not redundant. In this way, when a new pair is visited and it belongs to the list D of parity-1 pairs, we know that this pair is minimal and so belongs to the division core.

The algorithm effectively explores full set $\mathcal{I}_S$ and the bordering subset of $\mathrm{UpperClosure}(\mathrm{DivCore}_S)$ (in fact, among them, only the elements of $\mathrm{DivCore}_S$ are visited), which is at most $2n$ times larger. Note that all the predecessors of $(\underline{1}, \underline{0})$ and $(\underline{0}, \underline{1})$ are excluded. Let

$$\mathcal{I}_S^{\times} \coloneqq \{(u, v) \in \mathcal{I}_S \mid u \neq \underline{0}, v \neq \underline{0}\}.$$

Then, the algorithm performs at most $2n \left|\mathcal{I}_S^{\times}\right|$ iterations of the algorithm. Each iteration is dominated by an n-bit ParitySet computation together with its min-set (time $n2^n$). The total time complexity is upper bounded by $\mathcal{O}\left(\left|\mathcal{I}_S^{\times}\right| n^2 2^n\right)$. Note however that, due to maintaining the list D of parity-1 pairs, many visited pairs do not incur a parity set computation. In addition, by storing masks u and v for which the parity sets were already computed, we can avoid recomputing them for many pairs from $\mathcal{I}_S^{\times}$ as well. We conclude that the algorithm is expected to be much faster on practice.

The pseudocode is given in Algorithm 4.

5.4.1 Computing complete compact representation

Since the algorithm effectively enumerates full $\mathcal{I}_S^{\times}$, its max-set can be computed by marking redundant vectors during the enumeration (in addition, we need to manually add direct predecessors of $(\underline{0}, \underline{1})$ and $(\underline{1}, \underline{0})$ to avoid the enumeration of their exponentially-sized lower sets). For the compact modeling, it is left to compute $\mathrm{MinSet}(\mathcal{R}_S)$. For this purpose, we derive an alternative expression for $\mathcal{R}_S$.

Proposition 14. *Let $S : \mathbb{F}_2^n \to \mathbb{F}_2^M$. Then,*

$$\mathcal{R}_S = \bigcup_{(u,v) \in \mathrm{DivCore}_S} \{(u', v') \in \mathbb{F}_2^n \times \mathbb{F}_2^m \mid u' \succeq u, v' \succ v\}.$$

Algorithm 4 Computing division core of a bijection (BFS)

Input: lookup table $S \in (\mathbb{F}_2^n)^{2^n}$, lookup table $S^{-1} \in (\mathbb{F}_2^n)^{2^n}$
Output: $\mathrm{DivCore}_S \subseteq \mathbb{F}_2^n \times \mathbb{F}_2^n$
Complexity: $\mathcal{O}\left(\left|\mathcal{I}_S^\times\right| n^2 2^n\right)$, where $\mathcal{I}_S^\times := \{(u,v) \in \mathcal{I}_S \mid u \neq \underline{0}, v \neq \underline{0}\}$

$K \leftarrow \{(e_i, e_j) \mid i \in \{0,\dots,n-1\}, j \in \{0,\dots,n-1\}\}$ ▷ unknown parity vectors
$C \leftarrow \{(u,v) \mapsto 0 \mid u, v \in \mathbb{F}_2^n\}$ ▷ neighbor counts
$D \leftarrow \{(\underline{1}, \underline{0}), (\underline{0}, \underline{1})\}$ ▷ division core
$P \leftarrow D$ ▷ parity-one vectors
while K is not empty **do**
  $(u,v) \leftarrow \arg\min_{(u,v)\in K} (\min(\mathbf{wt}(u), \mathbf{wt}(v)), \max(\mathbf{wt}(u), \mathbf{wt}(v)))$
  Without loss of generality, assume that minimum is achieved in v
  delete (u,v) from K
  if $(u,v) \in P$ **then**
    $D \leftarrow D \cup \{(u,v)\}$
  else
    $U \leftarrow \neg\mathrm{MaxSet}\left(\mathrm{Supp}_{\mathrm{ANF}}\left(S^v\right)\right)$
    $P \leftarrow P \cup (U \times \{v\})$
    if $(u,v) \notin P$ **then**
      for all (u',v') a direct successor of (u,v) **do**
        $C((u',v')) \leftarrow C((u',v')) + 1$
        if $C((u',v'))$ is correct (†) **then**
          $K \leftarrow K \cup \{(u,v')\}$
return D

† Correctness of $C((u,v))$ means that $C((u,v))$ is equal to the number of predecessors of (u,v), excluding ones having a full-zero half:

$$C((u,v)) = \mathbf{wt}(u) + \mathbf{wt}(v) - [\mathbf{wt}(u) = 1] - [\mathbf{wt}(v) = 1]$$

Proof. Each set in the union defines redundant vectors identified by an element $(u, v) \in \mathrm{DivCore}_S$. Conversely, each redundant vector must have an associated irredundant vector from $(u, v) \in \mathrm{DivCore}_S$. □

It follows that $\mathrm{MinSet}(\mathcal{R}_S)$ can be computed from $\mathrm{DivCore}_S$ by replacing each vector $(u, v) \in \mathrm{DivCore}_S$ by the set of vectors (u, v'), where v' is taken from direct successors of v (i.e., $v' \succeq v, \mathbf{wt}(v') = \mathbf{wt}(v) + 1$). However, redundant vectors may occur there and a final computation of MinSet is needed. Assuming sparse $\mathrm{DivCore}_S$, it makes sense to use the naive quadratic MinSet algorithm instead of the dense one. The final complexity of computing $\mathrm{MinSet}(\mathcal{R}_S)$ is thus upper-bounded by $\mathcal{O}(|\mathrm{DivCore}_S|^2 \cdot n^2)$.

Corollary 4. *Let* $S : \mathbb{F}_2^n \to \mathbb{F}_2^m$. *Then,* $|\mathrm{MinSet}(\mathcal{R}_S)| \leq m \cdot |\mathrm{DivCore}_S|$.

Example 6. We ran the algorithm on a randomly generated 32-bit bijective S-box. Together with the generation and inversion, it took less than a core-day on a laptop with 64GB RAM. The resulting numbers are:

$$|\mathrm{DivCore}_S| = 7\,152, \quad |\mathrm{MaxSet}(\mathcal{I}_S)| = 2\,958, \quad |\mathrm{MinSet}(\mathcal{R}_S)| = 40\,093.$$

These numbers show that it would even be possible to model such an S-box in a cipher. Although it is unlikely that such a cipher would be of interest, this proof-of-concept show the power of the algorithm and of the compact representation to capture the simplicity of "heavy" S-boxes (i.e., the compactness of the maximal sets of monomials).

6 Application to LED

Derbez and Fouque [DF20] increased precision of traditional division property by two techniques: (1) computing "perfect" division property propagation tables of Super-Sboxes; (2) checking linear combinations of bits (inside Super-Sbox boundaries) at the input and at the output. In addition, the authors designed an ad-hoc search method, since modeling 16-bit S-boxes was not possible with state-of-the-art techniques. They considered lightweight block ciphers with 4-bit S-boxes and 16-bit Super-Sboxes, such as Midori64, Skinny-64, LED [GPPR11]. Their approach succeeded for Midori64 and Skinny-64, for which they improved best integral distinguishers by 1-2 rounds. However, the running time during their experiments with LED was not reasonable.

In this section, we apply our new framework to handle this case. The best integral distinguisher for LED is due to Hu, Wang and Wang [HWW20], who managed to model perfectly the MixColumn matrix of LED, which is MDS. The distinguisher covers 7 rounds, with 63 input active bits and full output state balanced. Full balanced state may hint towards possibility of weaker distinguishers (partially balanced state) on 8 or more rounds. We set to evaluate 8 rounds of LED using the two techniques by Derbez and Fouque implemented using our advancements. As we shall see, these two techniques are insufficient to find an 8-round integral distinguisher, if it exists.

All experiments were done on the version of LED with 128-bit key (the key size affects the constants in the Super-Sboxes).

6.1 Structure of LED and its model

The structure of LED is particularly convenient for our analysis. Each round consists of four standard operations: AddConstants(AC), SubBytes(SB), ShiftRows(SR), MixColumns(MC). The state of LED is a 4×4 array of 4-bit nibbles. The key is added only after every 4 rounds (a step).

The 8-round LED has a natural Super-Sbox decomposition: 4 rounds of Super-Sboxes ($\mathsf{SB} \to \mathsf{MC} \to \mathsf{AC}_{2i+1} \to \mathsf{SB}$, applied on columns) with the $\mathsf{SR} \to \mathsf{MC} \to \mathsf{SR}$ linear layers

in-between. For example, the following equation describes the Super-Sbox decomposition of the first two rounds (note that SR commutes with SB):

$$\begin{aligned} & \mathsf{AC}_0 \to \mathsf{SB} \to \mathsf{SR} \to \mathsf{MC} \to \mathsf{AC}_1 \to \mathsf{SB} \to \mathsf{SR} \to \mathsf{MC} \\ = \; & \mathsf{AC}_0 \to \mathsf{SR} \to (\mathsf{SB} \to \mathsf{MC} \to \mathsf{AC}_1 \to \mathsf{SB}) \to \mathsf{SR} \to \mathsf{MC}. \end{aligned}$$

The key addition happens outside of the Super-Sboxes and thus does not affect the modeling. However, the constant addition AC does affect Super-Sboxes, and we compute the division property transitions for each Super-Sbox separately, using the actual constant in the middle. In the following subsection, we describe modeling details for the two main components: Super-Sboxes and the MixColumns linear layer.

6.2 Modeling details

As our theoretical analysis shows that division property can be very naturally modeled by pure CNF formulas, we set to use a bare SAT-solver (not an SMT-solver). We chose Kissat [BFFH20], a recent solver which showed strong performance at a recent SAT competition [HJS+20].

We modeled 2 Super-Sbox rounds with $\mathsf{SR} \circ \mathsf{MC} \circ \mathsf{SR}$ layers in-between and outside (5 rounds of 16-bit maps with re-wirings in between). The missing 2 Super-Sbox rounds are treated by the linear mask analysis (Subsection 3.7) and by the trivial Super-Sbox transitions $\mathtt{1}^{16} \to \mathtt{1}^{16}$, $\mathtt{0}^{16} \to \mathtt{0}^{16}$. Each such model took less than a few minutes to solve on a laptop with an Intel(R) Core(TM) i5-10210U CPU and 64 GiB RAM.

Modeling MixColumn matrix The MixColumn matrix of LED is an MDS matrix M mapping $\mathbb{F}_{2^4}^4$ to itself. We apply directly our algorithms to compute the complementary lower and upper bounds on division property transitions. The lower bound (removing invalid transitions) consists of 33 412 vectors, the upper bound (removing redundant transitions) contains 334 974 429 vectors, the alternative upper bound contains 33 061 vectors. The total number of minimal transitions is 177 643 913. We observe that 33k clauses is reasonable for the lower bound. However, the upper bound is unnecessarily large. Therefore, we used the cardinality constraint described in Subsection 4.2 to remove $\mathcal{R}_M$ and used the 33k clauses to remove $\mathcal{I}_M$.

Modeling Super-Sbox We provide numbers for the case of Super-Sbox with the zero constant; the cases of other constants are similar. The division core contains 382 591 vectors and the number of valid minimal transitions is 8 481 417; the complementary lower and upper bounds contain 388 134 and 1 215 435 vectors respectively. These number are rather large, but still in a feasible range of modern SAT solvers. We used the 388 134 clauses to remove invalid trails precisely, while we used a cardinality bound to remove a part of redundant trails, to avoid using the 1 215 435 clauses per Super-Sbox for removing all redundant trails.

6.3 Exhausting all linear masks

We applied the approach from Subsection 3.7 to search for distinguishers with linear masks applied to an input and an output Super-Sbox.

Naive approach would be to exhaust all possible linear masks α, β and check the existence of respective distinguishers. However, as noticed by [DF20], many linear masks are redundant: an absence of distinguishers for one mask may imply absence of distinguishers for others, making them redundant (in case a distinguisher is found, redundant masks may be re-evaluated if needed).

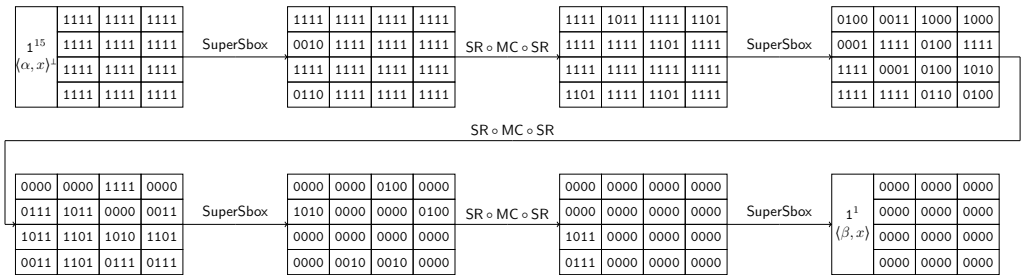

Figure 1: Example division trail from the 1st input Super-Sbox to the 1st output Super-Sbox. Covers input masks α such that the ANF of $\langle \alpha, \mathrm{SSB}_{0,0}^{-1}(x) \rangle$ contains a multiple of $x_4x_5x_7x_{12}x_{15}$ (zeroes in the first column after the first Super-Sbox), output masks β such that the ANF of $\langle \beta, \mathrm{SSB}_{3,0}(y) \rangle$ contains a multiple of $y_8y_{10}y_{11}y_{13}y_{14}y_{15}$ (ones in the first column before the last Super-Sbox).

On practice, many linear combinations turn to have the same set of maxterms in the ANF. For example, for the Super-Sbox of LED with the zero constant, the number of unique sets of maxterms among linear combinations of outputs is only 1785 (out of 65 535). The first step is thus to remove masks with duplicate sets of ANF maxterms.

From Theorem 3 it is clear that a mask is redundant if the lower closure of the respective ANF (i.e., that of $\langle \alpha, S_{\text{in}}^{-1} \rangle$ or $\langle \beta, S_{\text{out}} \rangle$) covers the lower closure of the ANF of another mask. As a result, we only need to consider masks corresponding to *minimal* by inclusion lower closures of the ANF. In the example constant-0 Super-Sbox of LED, the 1785 maxterm-unique ANFs reduce further to 255 (by a pairwise comparison). For the Super-Sbox' inverse, among 2021 maxterm-unique combinations again only 255 are minimal by (lower closure) inclusion.

Still, a straightforward search (as done in [DF20] for other ciphers) would require solving $16 \times 255 \times 255 \approx 1$ million (4×4 combinations of input and output Super-Sboxes) of search instances. This may be a feasible goal but it would require a significant computational effort. We describe a natural optimization that shows to be particularly helpful in the case of LED.

6.3.1 Reusing trails

Usually, one may expect that many linear combinations of output bits have similar ANFs. Therefore, a trail $\neg u \xrightarrow{F} v$ satisfying conditions of Theorem 3 for a pair of masks (α, β), may satisfy the conditions for some other pairs of masks (α', β') as well, even if both pairs correspond to unique and non-redundant ANFs. This condition can be checked much faster than solving a SAT instance. This suggests the following optimization: before solving the SAT instance for a pair of masks (α, β), check whether any previously found trail satisfies the condition.

This approach works well for the 8-round LED. For each combination of input/output Super-Sbox, about 30 trails are sufficient to show that the Super-Sbox model of 8-round LED does not allow to find integral distinguishers. All computed trails are provided in the code repository of the paper. An example trail is provided in Figure 1.

6.4 Summary

Using the described techniques, we managed to show that integral distinguishers for the 8-round LED (and, by Proposition 6, for its inverse), if any exists, can not be found using traditional bit-based division property even with perfect Super-Sbox modeling and arbitrary linear masks applied to Super-Sboxes at the input and at the output. To do this, we found a small set of division trails through 8-round LED that, together with Theorem 3, proves the claim. These trails are provided in the code repository of the paper.

References

[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 Transactions on Symmetric Cryptology*, 2017(2):99–129, 2017. 30

[BC13] Christina Boura and Anne Canteaut. On the Influence of the Algebraic Degree of F^{-1} on the Algebraic Degree of G ∘ F. *IEEE Transactions on Information Theory*, 59(1):691–702, 2013. 3, 10

[BC16] Christina Boura and Anne Canteaut. Another view of the division property. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 654–682. Springer, Heidelberg, 2016. 3, 6, 7, 8, 10

[BC20] Christina Boura and Daniel Coggia. Efficient MILP modelings for Sboxes and linear layers of SPN ciphers. *IACR Transactions on Symmetric Cryptology*, 2020(3):327–361, 2020. 16, 31

[BFFH20] Armin Biere, Katalin Fazekas, Mathias Fleury, and Maximillian Heisinger. CaDiCaL, Kissat, Paracooba, Plingeling and Treengeling entering the SAT Competition 2020. In *Proc. of SAT Competition 2020 – Solver and Benchmark Descriptions*, volume B-2020-1 of *Department of Computer Science Report Series b*, pages 51–53. University of Helsinki, 2020. 25

[BK16] Achiya Bar-On and Nathan Keller. A 2^{70} Attack on the Full MISTY1. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016*, pages 435–456, 2016. 3

[BKP16] Alex Biryukov, Dmitry Khovratovich, and Léo Perrin. Multiset-algebraic cryptanalysis of reduced Kuznyechik, Khazad, and secret SPNs. *IACR Trans. Symm. Cryptol.*, 2016(2):226–247, 2016. 7

[Car20a] Claude Carlet. Graph Indicators of Vectorial Functions and Bounds on the Algebraic Degree of Composite Functions. *IEEE Transactions on Information Theory*, 66(12):7702–7716, December 2020. 3, 4, 6, 14

[Car20b] Claude Carlet. Handling Vectorial Functions by Means of Their Graph Indicators. *IEEE Transactions on Information Theory*, 66(10):6324–6339, October 2020. 14

[CXZZ21] Siwei Chen, Zejun Xiang, Xiangyong Zeng, and Shasha Zhang. On the relationships between different methods for degree evaluation. *IACR Transactions on Symmetric Cryptology*, 2021(1):411–442, Mar. 2021. 3

[DF20] Patrick Derbez and Pierre-Alain Fouque. Increasing precision of division property. *IACR Transactions on Symmetric Cryptology*, 2020(4):173–194, 2020. 4, 12, 13, 21, 24, 25, 26

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer-Verlag, 2002. 3

[EKKT19] Zahra Eskandari, Andreas Brasen Kidmose, Stefan Kölbl, and Tyge Tiessen. Finding integral distinguishers with ease. In Carlos Cid and Michael J. Jacobson Jr:, editors, *SAC 2018*, volume 11349 of *LNCS*, pages 115–138. Springer, Heidelberg, 2019. 17

[GAB+20] Gerald Gamrath, Daniel Anderson, Ksenia Bestuzheva, Wei-Kun Chen, Leon Eifler, Maxime Gasse, Patrick Gemander, Ambros Gleixner, Leona Gottwald, Katrin Halbig, Gregor Hendel, Christopher Hojny, Thorsten Koch, Pierre Le Bodic, Stephen J. Maher, Frederic Matter, Matthias Miltenberger, Erik Mühmer, Benjamin Müller, Marc E. Pfetsch, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Christine Tawfik, Stefan Vigerske, Fabian Wegscheider, Dieter Weninger, and Jakob Witzig. The SCIP Optimization Suite 7.0. Technical report, Optimization Online, 2020. 16

[GD21] Shibam Ghosh and Orr Dunkelman. Security of sequential multiple encryption. In Patrick Longa and Carla Ràfols, editors, *Progress in Cryptology – LATINCRYPT 2021*, Berlin, Heidelberg, 2021. Springer Berlin Heidelberg. To appear. 3, 17

[GPPR11] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED block cipher. In Bart Preneel and Tsuyoshi Takagi, editors, *CHES 2011*, volume 6917 of *LNCS*, pages 326–341. Springer, Heidelberg, 2011. 18, 24

[GRW16] Faruk Göloğlu, Vincent Rijmen, and Qingju Wang. On the division property of S-boxes. Cryptology ePrint Archive, Report 2016/188, 2016. 3, 7

[HJS+20] M Heule, M Jarvisalo, M Suda, M Iser, T Balyo, and N Froleyks. SAT Competition 2020 - Results. https://satcompetition.github.io/2020/index.html, 2020. 25

[HLLT20] Phil Hebborn, Baptiste Lambin, Gregor Leander, and Yosuke Todo. Lower bounds on the degree of block ciphers. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part I*, volume 12491 of *LNCS*, pages 537–566. Springer, Heidelberg, 2020. 3

[HLLT21] Phil Hebborn, Baptiste Lambin, Gregor Leander, and Yosuke Todo. Strong and tight security guarantees against integral distinguishers. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021*. Springer-Verlag, 2021. To appear. 3

[HLM+20] Yonglin Hao, Gregor Leander, Willi Meier, Yosuke Todo, and Qingju Wang. Modeling for three-subset division property without unknown subset - improved cube attacks against Trivium and Grain-128AEAD. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 466–495. Springer, Heidelberg, 2020. 3, 4

[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,

ASIACRYPT 2020, Part I, volume 12491 of *LNCS*, pages 446–476. Springer, Heidelberg, 2020. 3, 4

[HWW20] Kai Hu, Qingju Wang, and Meiqin Wang. Finding bit-based division property for ciphers with complex linear layer. *IACR Trans. Symm. Cryptol.*, 2020(1):396–424, 2020. 3, 11, 17, 18, 24

[LDF20] Baptiste Lambin, Patrick Derbez, and Pierre-Alain Fouque. Linearly equivalent S-boxes and the division property. *Designs, Codes and Cryptography*, 88(10):2207–2231, October 2020. 4, 12, 13

[Mat97] Mitsuru Matsui. New block encryption algorithm MISTY. In Eli Biham, editor, *FSE'97*, volume 1267 of *LNCS*, pages 54–68. Springer, Heidelberg, 1997. 3

[McC56] E. J. McCluskey. Minimization of boolean functions. *The Bell System Technical Journal*, 35(6):1417–1444, 1956. 16, 30

[MT21] Ashwini Kumar Malviya and Namita Tiwari. Quantum algorithm to identify division property of a multiset. *Arabian Journal for Science and Engineering*, 46(9):8711–8719, Sep 2021. 20

[Qui55] W. V. Quine. A way to simplify truth functions. *The American Mathematical Monthly*, 62(9):627–631, 1955. 16, 30

[SHZ+16] Bing Sun, Xin Hai, Wenyu Zhang, Lei Cheng, and Zhichao Yang. New observation on division property. *Science China Information Sciences*, 60(9):098102, December 2016. 3

[Sin05] Carsten Sinz. Towards an Optimal CNF Encoding of Boolean Cardinality Constraints. In *Principles and Practice of Constraint Programming - CP 2005*, volume 3709, pages 827–831. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. 17

[ST] 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, *SecITC 2017*, volume 10543 of *LNCS*, pages 150–165, Cham. Springer. 3

[SWW16] Ling Sun, Wei Wang, and Meiqin Q. Wang. MILP-aided bit-based division property for primitives with non-bit-permutation linear layers. *IET Information Security*, 14(1):12–20, 2016. 3

[SWW21] Ling Sun, Wei Wang, and Meiqin Wang. Accelerating the search of differential and linear characteristics with the sat method. *IACR Transactions on Symmetric Cryptology*, 2021(1):269–315, Mar. 2021. 17

[TM16a] Yosuke Todo and Masakatu Morii. Bit-based division property and application to simon family. In Thomas Peyrin, editor, *FSE 2016*, volume 9783 of *LNCS*, pages 357–377. Springer, Heidelberg, 2016. 3, 4, 6, 12

[TM16b] Yosuke Todo and Masakatu Morii. Compact representation for division property. In Sara Foresti and Giuseppe Persiano, editors, *CANS 16*, volume 10052 of *LNCS*, pages 19–35. Springer, Heidelberg, 2016. 9

[Tod15a] Yosuke Todo. Integral cryptanalysis on full MISTY1. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 413–432. Springer, Heidelberg, 2015. 3

[Tod15b] Yosuke Todo. Structural evaluation by generalized integral property. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 287–314. Springer, Heidelberg, 2015. 3, 6, 12

[Udo21] Aleksei Udovenko. MILP modeling of Boolean functions by minimum number of inequalities. Cryptology ePrint Archive, Report 2021/1099, 2021. 15

[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, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 648–678. Springer, Heidelberg, 2016. 3, 8, 21

[ZR18] Wenying Zhang and Vincent Rijmen. Division cryptanalysis of block ciphers with a binary diffusion layer. *IET Information Security*, 13(2):87–95, August 2018. 3, 11

A Framework - API example

We describe an excerpt of usage of our framework with a Python binding (the set manipulation part is written in C++), to show the ease of using and its power. In the example below, we manipulate large sets of 32-bit vectors and each manipulation takes at most a few seconds. Due to dense packing of bits, one such set occupies only 512MB of RAM. For more details, see https://github.com/CryptoExperts/AC21-DivProp-Convexity.

```
from subsets import DenseSet, DenseBox
from divprop import SboxDivision, Sbox
from divprop.ciphers import SSB_LED

sbox = SSB_LED().MC
divcore = sbox.graph_dense().Mobius().MaxSet().Not()
divcore
# <DenseSet hash=473e100dda56d8d9 n=32 wt=177643913 | 16:177643913>

dc = SboxDivision(sbox)
dc.divcore
# <DenseSet hash=473e100dda56d8d9 n=32 wt=177643913 | 16:177643913>
dc.invalid_max  # MaxSet(I_S)
# <DenseSet hash=6f370f1ff06cbe0a n=32 wt=33412 | 15:2596 16:5442 17:6876
#  18:6415 19:5108 20:3469 21:1896 22:942 23:423 24:161 25:68 26:15 27:1>
dc.redundant_min  # MinSet(R_S)
# <DenseSet hash=0fb28b3f6fdef9b4 n=32 wt=334974429 | 17:334974429>
dc.redundant_alternative_min  # MinSet(R'_S)
# <DenseSet hash=1940406c7b877628 n=32 wt=33061 | 5:8 6:11 7:31 8:163 9:418
# 10:964 11:2024 12:3404 13:4925 14:6240 15:6689 16:5474 17:2710>
```

B Quine-McCluskey algorithm for Boolean minimization

Boura and Coggia reformulated the Quine-McCluskey [Qui55, McC56] algorithm for minimization of Boolean Functions, previously used in [AST+17] to optimize modeling. They show that its first part consists in finding all maximal subsets of the form $a \oplus \text{LowerClosure}(u)$, $a, u \in \mathbb{F}_2^n$ (*prime implicants*) of the given set $P \subseteq \mathbb{F}_2^n$ (which is the complement of the set being modeled) and provide efficient yet heuristic algorithm for this

step. We design a simple more efficient algorithm in our framework with simple complexity analysis. Although it is inferior to an optimized implementation of the Quine-McCluskey algorithm, this implementation highlights the universality of our framework.

Similarly to the Boura-Coggia algorithm, our algorithm enumerates all $a \in P$ and finds all maximal u such that $a \oplus \text{LowerClosure}(u)$ is a subset of P. The problem naturally reduces to finding all maximal u such that $\text{LowerClosure}(u) \subseteq (a \oplus P)$. Let

$$U = \bigcup_{u:\text{LowerClosure}(u) \subseteq a \oplus P} \text{LowerClosure}(u)$$

be the lower set of all solutions. It is easy to show that its complementary upper set has only non-elements of P in its *min-set*. Indeed, otherwise such an element of P could be merged with U into a bigger set of solutions. This leads to the following algorithm: compute the upper set of *non-elements* of $a \oplus P$, and complement it to get U. Let $\widehat{U} = \text{MaxSet}(U)$. It is only left to remove elements of $\widehat{U}$ having common bits set with current value of a to ensure unique solutions and to couple the remaining vectors with the a. See Algorithm 5 for a details.

For example, the complements of the DDT supports of the S-boxes of AES and SKINNY128 are processed in 22s/36s respectively on a single core of a laptop using Algorithm 5. This drastically improves the reported 15m/2h by [BC20], although the improvement may also come from the use of C++ in our benchmark.

We remark that, on practice, the first step of the Quine-McCluskey algorithm is not dominating the overall complexity. Therefore, our improved algorithm does not immediately provide a significant improvement. However, coupled with, for example, the greedy algorithm for step 2, it allows to obtain high quality (although sub-optimal) models for larger S-Boxes.

Algorithm 5 Find all subsets of $P \subseteq \mathbb{F}_2^n$ of the form $a \oplus \text{LowerClosure}(u)$ with maximal u

Input: $P \subseteq \mathbb{F}_2^n$
Output: $S = \{a, u \in \mathbb{F}_2^n \mid (a \oplus \text{LowerClosure}(\{u\})) \subseteq P,\ u \text{ maximal such vector}\}$
Complexity: $\mathcal{O}(n2^n|P|)$

1: $S \leftarrow \emptyset$
2: **for all** $a \in P$ **do**
3: $\quad X \leftarrow \underline{a \oplus P}$
4: $\quad X \leftarrow \text{UpperClosure}\left(\overline{X}\right)$
5: $\quad U \leftarrow \text{MaxSet}(X)$
6: $\quad S \leftarrow S \cup \{(a, u) \mid u \in U, u \wedge a = \underline{0}\}$
7: **return** S

C (Multi-dimensional) cardinality bounds

In order to allow faster conflicts during the SAT/MILP search, we describe techniques of cardinality bounds on the transitions. The idea is to model an *over-approximation* of the set of minimal transitions by relatively small formulas. This allows solvers to quickly skip a large part of bad transitions, and to process the remaining precise constraints on the remaining smaller search space. In addition, part of precise constraints covered by approximate constraints becomes redundant and can be removed.

The one-dimensional case from Subsection 4.2 generalizes easily to higher dimensions.

Definition 13. Let $t_0, t_1, \ldots, t_{k-1} \in \mathbb{Z}_+$, $2n = \sum_i t_i$. Define the $(t_0, t_1, \ldots, t_{k-1})$-*projection* $\rho_{t_0,t_1,\ldots,t_{k-1}}$:

$$\begin{aligned}\rho_{t_0,t_1,\ldots,t_{k-1}} &: \mathbb{F}_2^{2n} \to \{0,\ldots,t_0\} \times \ldots \times \{0,\ldots,t_{k-1}\} \\ &: x \mapsto (\mathbf{wt}(x_{0,\ldots,t_0-1}), \mathbf{wt}(x_{t_0,\ldots,t_0+t_1-1}), \ldots, \mathbf{wt}(x_{n-t_{k-1},\ldots,n-1})).\end{aligned}$$

Let t^r-projection ρ_{t^r} be a shortcut for $\rho_{t,t,\ldots,t}$, where t is repeated r times.

Example 7. $\rho_{3^4}(\texttt{101 000 111 011}) \coloneqq \rho_{3,3,3,3}(\texttt{101 000 111 011}) = (2,0,3,2)$.

Note that the partial order framework (lower/upper/min-/max-/convex sets) applies naturally sets of the form $\{0,\ldots,t_0\} \times \ldots \times \{0,\ldots,t_{k-1}\}$.

Proposition 15. *Let* $t, r \in \mathbb{Z}_+$, $2n = t \cdot r$, $\mathbf{M} \coloneqq \rho_{t^r}(\mathcal{M}_S)$,

$$\begin{aligned}L &\coloneqq \mathrm{MaxSet}\left(\overline{\mathrm{UpperClosure}\left(\mathbf{M}\right)}\right), \\ H &\coloneqq \mathrm{MinSet}\left(\overline{\mathrm{LowerClosure}\left(\mathbf{M}\right)}\right).\end{aligned}$$

Then, for all $w \in \mathcal{M}_S$, *we have*

$$\begin{aligned}&\rho_{t^r}(w) \not\preceq l, \quad \forall l \in L, \\ &\rho_{t^r}(w) \not\succeq h, \quad \forall h \in H.\end{aligned}$$

Remark 14. The case $t = 2n, r = 1$ corresponds to the one-dimensional cardinality bound described above. However, for higher dimensions, the bound becomes "collective", i.e. a system of constraints. The opposite edge case, $t = 1, r = 2n$, partially corresponds to our general modeling by removing sets $\mathcal{I}_S, \mathcal{R}_S$. While indeed $L = \mathrm{MaxSet}\left(\mathcal{I}_S\right)$ in this case, it only holds that $\mathrm{UpperClosure}\left(H\right) \supseteq \mathcal{R}_S$.

Example 8. Consider the Super-Sbox of LED (with the zero constant in the middle, see Section 6). For $t = 2n, r = 1$, for all $w \in \mathcal{M}_S$, we get

$$\begin{aligned}&L_1 = \{(5)\}, H_1 = \{(17)\}, \\ &\mathbf{wt}(w) \not\preceq l, \quad \forall l \in L_1, \\ &\mathbf{wt}(w) \not\succeq h, \quad \forall h \in H_1.\end{aligned}$$

For $t = n, r = 2$, for all $(u, v) \in \mathcal{M}_S$, we get

$$\begin{aligned}&L_2 = \{(0,15),(1,6),(2,3),(3,2),(6,1),(15,0)\}, \\ &H_2 = \{(1,14),(2,13),(5,10),(7,9),(9,6),(11,5),(13,3),(14,2),(16,1)\}, \\ &(\mathbf{wt}(u), \mathbf{wt}(v)) \not\preceq l, \quad \forall l \in L_2 \\ &(\mathbf{wt}(u), \mathbf{wt}(v)) \not\succeq h, \quad \forall h \in H_2.\end{aligned}$$

Observe that, in the example above, the 1-dimensional bound covers some elements from the 2-dimensional bound: $(2,3),(3,2) \in L_2$ are redundant as they are covered by $(5) \in L_1$; $(16,1) \in H_2$ is redundant as it is covered by $(17) \in H_1$. For more fine-grained projections however, we did not observe a significant reduction on practice.

Finally, we show how a constraint $\rho_{t^r} \not\preceq l$ can be encoded.

Proposition 16. *The constraint* $\rho_{t^r}(w) \not\preceq l$, $l \in \mathbb{Z}_+^r$, *can be encoded by a single CNF clause given the* r *cardinality vectors* $c_0, \ldots, c_{r-1}; c_i \coloneqq \mathbf{wt}(w_{it,\ldots,it+t-1})$ *are encoded, as follows:*

$$\bigvee_i c_i \geq l_i + 1.$$

Remark 15. Here, by the *cardinality vector* we mean the unary-encoded sum of the variables.

Remark 16. MILP system can be generated similarly. The unary cardinality vector $(c_{i,j})$ (with the binary variable $c_{i,j} \in \{0,1\}$ equal to 1 if and only if $c_i \geq j$) can be encoded as $j \cdot c_{i,j} \leq c_i \leq j - 1 + (t - j + 1) \cdot c_{i,j}$ for all constant j.

Example 9. Recall the example from above of the LED's Super-Sbox. For the record, there are in total 39 704 319 invalid and 4 246 781 560 redundant vectors (out of 2^{32} possible vectors).

Using the 1-dimensional cardinality bound (1 cardinality vector of length 32) and 1+1 extra clauses, 242 825 invalid (0.61%) and 1 846 943 453 (43%) redundant vectors are removed.

Using the 2-dimensional cardinality bound (2 cardinality vectors of length 16) and 6+9 extra clauses, 756 157 invalid (1.9%) and 2 402 421 469 (57%) redundant vectors are removed.

Using the 8-dimensional cardinality bound (8 cardinality vectors of length 4) and 341+4 478 extra clauses, 33 861 085 invalid (85%) and 4 170 189 661 (98%) redundant vectors are removed.

Finally, among 388 134 vectors from MaxSet $(\mathcal{I}_S)$, still 329 670 vectors remain to be removed to completely avoid invalid trails.

We remark that in the case of LED from Section 6 we did not observe a noticeable performance improvement from using multidimensional cardinality bounds. The tool may be however useful in other, heavier cases.