

軽量ブロック暗号 Piccolo の MILP 手法を用いた Division Property の検証

佐藤 寛樹^{1,a)} 三村 守^{1,b)} 田中 秀磨^{1,c)}

概要: Division Property (DP) は Integral Property を一般化したものである。また、より複雑な探索を可能にするため Mixed Integer Linear Programming (MILP) を応用した手法が提案されている。本論文では 64bit 軽量ブロック暗号 Piccolo に対する久保らによる DP 探索と高階差分特性との比較の検証を、MILP を用いて行う。その結果、8, 12, 48 次の評価の妥当性が確認できた。24, 32 次に関しては久保の劣化に関する検証の正当性と SizeReduce を使用しない MILP の有効性が確認できた。さらに 63 次に関しては既存結果より 1 段多い 7 段 DP を発見できた。

キーワード: Piccolo, MILP, Division Property, 高階差分特性

Analysis of Division Property with MILP method of Lightweight Blockcipher Piccolo

HIROKI SATO^{1,a)} MAMORU MIMURA^{1,b)} HIDEMA TANAKA^{1,c)}

Abstract: Integral property has been generalized by the division property. The Mixed Integer Linear method has been proposed to enable more effective search. In this paper, using MILP, we verify the comparison of division property and the higher-order differential property on 64 bit lightweight blockcipher Piccolo by Kubo et al. As the result, the validity of the 8th, 12th and 48th evaluations are confirmed. Regarding the 24th and 32nd evaluation, the validity of the verification concerning the analysis of Kubo and the effectiveness of MILP without SizeReduce are confirmed. Regarding the 63rd evaluation, we can find 7-round DP which is one round higher than existing result.

Keywords: Piccolo, MILP, Division Property, Higher-order differential property

1. はじめに

藤堂は Integral Property を一般化した DP を提案している [1]。DP を用いた手法により、CRYPTO2015 において MISTY1 のフルラウンド解読が可能なが報告されている [2]。しかし、DP を用いた手法は非常に大きいメモリを必要とし、現実的な計算時間の観点から 32 次以上の DP の解析が困難であるという問題があった。より詳細

で汎用な暗号の探索を可能にするため、ASIACRYPT2016 において Xiang らは MILP を応用した手法を提案している [3]。また、非ビット順列の線形層にも適用できるように Ling らにより MILP 手法は改良されている [4]。

Piccolo[5] は CHES2011 において SONY の渋谷らが提案した 64bit 軽量ブロック暗号であり、久保らにより藤堂の手法を用いて DP と高階差分特性との比較が行われている [6]。しかし、久保らは DP の伝搬規則上、高階差分に比べて特性が劣化する場合を確認しており、この発生要因として SizeReduce を使用していることを指摘している。

本稿では Piccolo の DP 探索において Ling らの MILP 手法を適用し、久保らの結果を検証する。

¹ 防衛大学校情報工学科
National Defence Academy in Japan
a) em57032@nda.ac.jp
b) mim@nda.ac.jp
c) hidema@nda.ac.jp

2. 軽量ブロック暗号 Piccolo

Piccolo の全体構造を図 1 に示す。

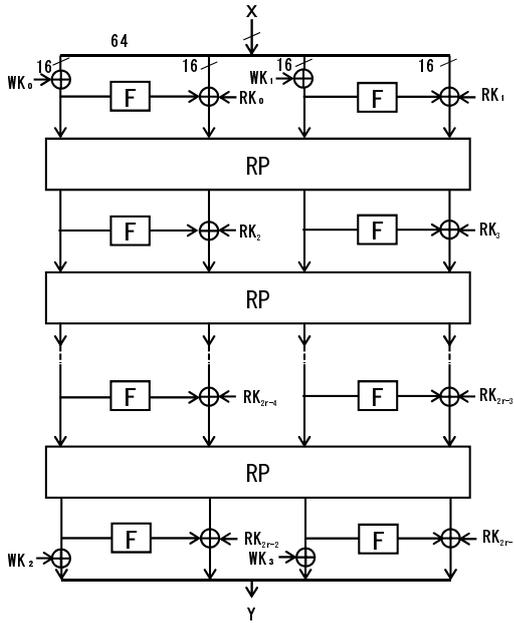


図 1 Piccolo の全体構造

Piccolo は変形 Feistel 構造を持つ。Piccolo は入力 64bit を 16bit ずつに分割し、入出力 16bit の F 関数を 2 つ並列に用いて 32bit ずつ処理を行う。平文から暗号文への処理は、まず 1 段目の F 関数入力においてホワイトニングキー (WK) による鍵加算を行う。次にラウンド転置 (RP) を行いラウンド出力とする。最終段の F 関数出力において再び WK による鍵加算を行った後、暗号文として出力する。F 関数の内部構造を図 2 に示す。

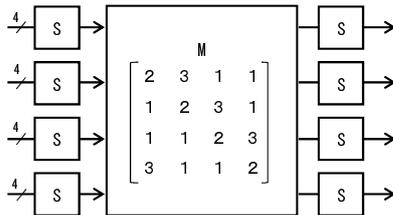


図 2 F 関数の構造

F 関数は 16bit の入力を 4bit に分割し 4bit 入出力の S-Box に 4 並列に入力する。その出力を行列 M に基づいて拡散し、その出力が再度 S-Box へ入力される構造となっている。行列 M の詳細は第 4.1 節で示す。表 2 に F 関数内で使用する S-Box を 16 進数表記で示す。この S-Box の代数次数は 3 であることは、久保らによって示されている [6]。

RP では 8bit 単位でデータの入れ替えを行う。RP の内部構造を図 3 に示す。

表 1 S-Box

入力	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
出力	e	4	b	2	3	8	0	9	1	a	7	f	6	c	5	d

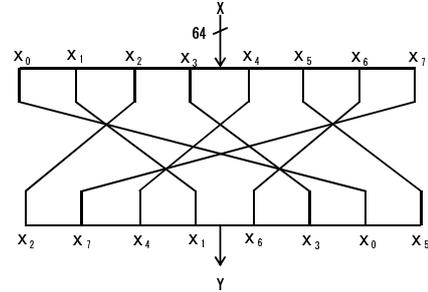


図 3 ラウンド転置 (RP)

3. 関連研究 [1] [2]

Integral Property はビット積関数 $\pi_u(x)$ と Division Property D_k^n を用いて一般化される。

3.1 表記法

本稿で用いる表記法を示す。 $x[i]$ は n bit 入力 $x \in \mathbb{F}_2^n$ の i 番目のビットを表す ($i = 1, 2, \dots, n$)。ハミング重み $wt(x)$ は $wt(x) = \sum_{i=1}^n x_i$ で与えられる。任意の集合 \mathbb{K} について、 $|\mathbb{K}|$ は \mathbb{K} の要素数を表す。また、 ϕ は空集合を表す。 m 次元ベクトル $\vec{x} \in (\mathbb{F}_2^{l_0} \times \mathbb{F}_2^{l_1} \times \dots \times \mathbb{F}_2^{l_{m-1}})$ のハミング重みベクトルを $Wt(\vec{x})$ と表し、 $Wt(\vec{x}) = (wt(x_1), wt(x_2), \dots, wt(x_m)) \in \mathbb{Z}^m$ とする。任意の m 次元ベクトル $\vec{k} \in \mathbb{Z}^m$ と $\vec{k}' \in \mathbb{Z}^m$ において、すべての $i (= 1, 2, \dots, m)$ において $k_i \geq k'_i$ ならば $k \succeq k'$ と表記し、そうでなければ $k \not\succeq k'$ と表記する。

3.2 ビット積関数

定義 1 【ビット積関数】

任意の n bit 変数 $u \in \mathbb{F}_2^n$ に対するビット積関数 $\pi_u(x)$ の定義は次のとおりである。ただし入力 x は n bit, 出力 $\pi_u(x)$ は 1bit である。

$$\pi_u(x) := \prod_{i=0}^{n-1} x[i]^{u[i]}$$

任意の m 次元ベクトル $\vec{u} \in (\mathbb{F}_2^{n_0} \times \mathbb{F}_2^{n_1} \times \dots \times \mathbb{F}_2^{n_{m-1}})$ に対するビット積関数 $\pi_{\vec{u}}(\vec{x})$ の定義は次の通りである。ここで x は $x \in (\mathbb{F}_2^{n_0} \times \mathbb{F}_2^{n_1} \times \dots \times \mathbb{F}_2^{n_{m-1}})$ である。

$$\pi_{\vec{u}}(\vec{x}) := \prod_{i=0}^{n-1} \pi_{u_i}(x_i)$$

ブール関数の ANF (Algebraic Normal Form) を用いれば $f: \mathbb{F}_2^n \rightarrow \mathbb{F}_2$ は次式で表現される。ただし、 $a_u^f \in \{0, 1\}$ は

f と u による係数である.

$$f(x) = \bigoplus_{u \in \mathbb{F}_2^n} a_u^f \left(\prod_{i=1}^n x[i]^{u[i]} \right) = \bigoplus_{u \in \mathbb{F}_2^n} a_u^f \pi_u(x)$$

3.3 Division Property

DP は以下のように定義される.

定義 2 【 Division Property 】

m 次元ベクトル $\vec{a} \in (\mathbb{F}_2^{l_0} \times \mathbb{F}_2^{l_1} \times \dots \times \mathbb{F}_2^{l_{m-1}})$ の多重集合を \mathbb{X} と表す. 次式を満たすとき多重集合 \mathbb{X} は DP $D_{\mathbb{K}}^{l_0, l_1, \dots, l_{m-1}}$ を持つ.

$$\bigoplus_{x \in \mathbb{X}} \pi_u(x) = \begin{cases} \text{unknown} & \text{if there is } \vec{k} \in \mathbb{K} \text{ s.t. } \text{Wt}(\vec{u}) \succeq \vec{k} \\ 0 & \text{otherwise} \end{cases}$$

ただし, \mathbb{K} は m 次元ベクトル \vec{k} の集合であり, i 番目の要素 k_i は $0 \sim l_i$ の値をとる.

3.3.1 Bit-Based Division Property の伝搬規則

藤堂が示している DP の伝搬規則は, *Substitution*, *Copy*, *XOR*, *Split* 及び *Concatenation* の 5 つがある. 本節では, MILP 手法において *Copy* 及び *XOR* のみ使用して線形不等式を生成するため, Bit-Based DP で使用する *Copy*(図 4(a)) 及び *XOR*(図 4(b)) についてのみ述べる.

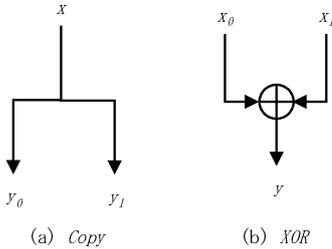


図 4 DP の伝搬規則

伝搬規則 1 【 *Copy* 】

Copy は入力 $x \in \mathbb{F}_2$ を図 4(a) に示すように $(y_0, y_1) = (x, x)$ と計算する関数である. \mathbb{X} と \mathbb{Y} をそれぞれ入力 x , 出力 y の多重集合とする. 入力多重集合 \mathbb{X} が DP $D_{\{k\}}^1$ を持つとすると, 出力多重集合は DP $D_{\mathbb{K}'}^{1 \times 1}$ を持つ. \mathbb{K}' は以下の 2 つの場合がある.

$$\begin{cases} \mathbb{K}' = \{(0, 0)\} & \text{if } k = 0 \\ \mathbb{K}' = \{(0, 1), (1, 0)\} & \text{if } k = 1 \end{cases}$$

伝搬規則 2 【 *XOR* 】

XOR は入力 $x = (x_0, x_1) \in \mathbb{F}_2 \times \mathbb{F}_2$ を図 4(b) に示すように $y = x_0 \oplus x_1$ と計算する関数である. \mathbb{X} と \mathbb{Y} をそれぞれ入力 x , 出力 y の多重集合とする. 入力多重集合 \mathbb{X} が

DP $D_{\{\vec{k}\}}^{1 \times 1}$ を持つとすると, 出力多重集合は DP $D_{\mathbb{K}'}^1$ を持つ. \mathbb{K}' は以下の 3 つの場合がある.

$$\begin{cases} \mathbb{K}' = \{(0)\} & \text{if } \vec{k} = (0, 0) \\ \mathbb{K}' = \{(1)\} & \text{if } \vec{k} = (1, 0) \text{ or } (0, 1) \\ \mathbb{K}' = \emptyset & \text{if } \vec{k} = (1, 1) \end{cases}$$

3.3.2 MILP 手法の Bit-Based Division Property[3][4]

Bit-Based DP は Integral distinguisher を探すための有効な手段ではあるが, 実行に必要なメモリ量と計算時間から 32 次以上のものは探索することが困難である. そのため, ASIACRYPT2016 において Xiang らにより MILP 手法を用いた DP (以下 MILP 手法) が提案された. この手法は, DP の伝搬規則を線形不等式に置き換えることで Gurobi[7] などの MILP オプティマイザを有効に適用するものである. この結果, Bit-Based DP と比較して計算コストを大幅に減少させ, 32 次以上の探索を可能にする.

定義 3 【 Division Trail 】

F をブロック暗号のラウンド関数とする. ブロック暗号の入力多重集合が DP $D_{\vec{k}}^{1^n}$ を持つとすると, i 回のラウンド関数 F の繰り返しにより伝搬した後の DP は $D_{\mathbb{K}_i}^{1^n}$ によって表される. したがって, 次のような DP 伝搬の連鎖が得られる.

$$\{\vec{k}\} := \mathbb{K}_0 \xrightarrow{F} \mathbb{K}_1 \xrightarrow{F} \mathbb{K}_2 \xrightarrow{F} \dots$$

ここで, 任意のベクトル $\vec{k}_i^* \in \mathbb{K}_i (i \geq 1)$ において, k_{i-1}^* が k_i^* に DP の伝搬規則に従って伝搬するには, $k_{i-1}^* \in \mathbb{K}_{i-1}$ が存在することが条件となる. この条件を満たし, $(k_0, k_1, \dots, k_r) \in \mathbb{K}_0 \times \mathbb{K}_1 \times \dots \times \mathbb{K}_r$ において k_{i-1}^* がすべての $i \in \{1, 2, \dots, r\}$ において k_i に伝搬するとき, (k_0, k_1, \dots, k_r) を r ラウンドの *Division Trail* と呼ぶ.

定理 1 ブロック暗号の入力多重集合が DP $D_{\vec{k}}^{1^n}$ を持ち, F をラウンド関数とすると r ラウンド DP 伝搬は次のように表される.

$$\{\vec{k}\} := \mathbb{K}_0 \xrightarrow{F} \mathbb{K}_1 \xrightarrow{F} \mathbb{K}_2 \xrightarrow{F} \dots \xrightarrow{F} \mathbb{K}_r$$

つまり \vec{k} から始まるすべての r ラウンド *Division Trail* の最後のベクトルのセットは \mathbb{K}_r と等しくなる.

次に伝搬規則 1, 2 で示した *Copy*, *XOR* 及び S-Box の MILP 手法で使用するビット単位の線形不等式の Model を示す.

Model 1 【 *Copy* 】

Copy の *Division Trail* を (a) $\xrightarrow{\text{Copy}}$ (b_0, b_1, \dots, b_m) とすると *Copy* の DP 伝搬は次の線形不等式で表現される.

$$a - b_0 - b_1 - \dots - b_m = 0$$

ただし a, b_0, b_1, \dots, b_m はバイナリ値である。

Model 2 【XOR】

XOR の *Division Trail* を $(a_0, a_1, \dots, a_m) \xrightarrow{XOR} (b)$ とすると XOR の DP 伝搬は次の不等式で表現されている。

$$a_0 + a_1 + \dots + a_m - b = 0$$

ただし a_0, a_1, \dots, a_m, b はバイナリ値である。

Modelling 【S-Box】

n bit S-Box の *Division Trail* は $\{0, 1\}^{2n} \in \mathbb{R}^{2n}$ 上の $2n$ 次元のベクトルとして表現できる。つまり *Division Trail* は $\{0, 1\}^{2n}$ の部分集合 P からなる。次に, *Sage software* の *inequality-generator()* を使用して凸包 P の H 表現 (half-space representation) を計算し, 線形不等式 L のセットを得る [8]。しかし, そのまま適用すると MILP を実行するには L が多すぎるので, Sun らによって提案された Greedy Algorithm を利用して削減する [9]。その結果得られた L で S-Box の DP 伝搬を表す不等式群が決定できる。

3.3.3 Initial Division Property

定理 2 $(a_0^0, a_1^0, \dots, a_{n-1}^0) \rightarrow \dots \rightarrow (a_0^r, a_1^r, \dots, a_{n-1}^r)$ を r ラウンド *Division Trail* とすると L は変数 $a_i^j (i = 0, 1, \dots, n-1, j = 0, 1, \dots, r)$ で定義される。

初期入力の DP が $D_{\{\vec{k}\}}^{1^n} (\vec{k} = (k_0, k_1, \dots, k_{n-1}))$ とすると, L に $a_i^0 = k_i (i = 0, 1, \dots, n-1)$ を加える必要があり, 初期入力の L の実行可能解はベクトル \vec{k} からなる *Division Trail* である。

3.3.4 Stopping Rule

\mathbb{X} が DP $D_{\mathbb{K}}^{1^n}$ を持つ多重集合であり, \mathbb{X} が \mathbb{K} にすべての n 個の単位ベクトルを含むとき *Integral Property* を持たないものとする。 i ラウンド暗号化された後の出力 DP を $D_{\mathbb{K}_i}^{1^n}$ とすると, 入力 DP は $D_{\mathbb{K}_0}^{1^n}$ で表される。 \mathbb{K}_{k+1} のとき初めて n 個のベクトルすべてが含まれたとすると, DP 伝搬は止まり $D_{\mathbb{K}_r}^{1^n}$ から r ラウンド distinguisher が得られる。この考えを基にすると \mathbb{K}_r にすべての単位ベクトルが含まれているかどうかだけ確認する必要がある。定理 1 によると \mathbb{K}_r 内のすべてのベクトルを調べることは r ラウンド *Division Trail* の最後のベクトルを調べることと同義である。 $(a_0^0, a_1^0, \dots, a_{n-1}^0) \rightarrow \dots \rightarrow (a_0^r, a_1^r, \dots, a_{n-1}^r)$ を r ラウンド *Division Trail* とし L を与えられた入力 DP からなるすべての *Division Trail* が実行可能解である線形不等式とすると目的関数は以下のように導出できる。

$$Obj : \text{Min}\{a_0^r + a_1^r + \dots + a_{n-1}^r\}$$

3.4 複雑な線形層に関するモデル

M^{PR} を $n \times n$ 行列とし, 線形層のプリミティブ表現とする。この行列は次のように表される。ただし, $m_{i,j} \in \{0, 1\}$ である。

$$M^{PR} = \begin{pmatrix} m_{0,0} & m_{0,1} & \dots & m_{0,n-1} \\ m_{1,0} & m_{1,1} & \dots & m_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n-1,0} & m_{n-1,1} & \dots & m_{n-1,n-1} \end{pmatrix}$$

行列 M^{PR} の i 列目のハミング重みを c_i , j 行目のハミング重みを r_j とする。 $c_M = \sum_{i=0}^{n-1} c_i (= \sum_{j=0}^{n-1} r_j)$ を M^{PR} 上のゼロを含まない個数とする。

上記のような行列 M^{PR} において入力が拡散されるとき i 番目の入力値は c_i 回 Copy される。また M^{PR} に拡散された後の j 番目の出力値は r_j 回 XOR されている。

したがってこれらの Copy の DP を表すために中間変数 c_i を導入する必要がある。すべての入力ビットに対して DP の伝搬を表すとき, 中間変数 c_M は $t_0 \sim t_{c_M-1}$ が必要となる。

M^{PR} の入力が DP $D_{\{\vec{x}\}}^{1^n} (\vec{x} = (x_0, x_1, \dots, x_{n-1}))$ を持ち, 出力が DP $D_{\{\vec{y}\}}^{1^n} (\vec{y} = (y_0, y_1, \dots, y_{n-1}))$ を持つとする。中間変数 c_0 を x_0 に割り当て, 次の変数 c_1 を x_1 に, などと逐次的に割り当てていくとする。Model 1 よりすべての入力の Copy は次の線形不等式で表すことができる。

$$\begin{cases} x_0 - t_0 - t_1 - \dots - t_{c_0-1} = 0 \\ x_0 - t_{c_0} - t_{c_0+1} - \dots - t_{c_0+c_1-1} = 0 \\ \vdots \\ x_{n-1} - t_{c_M-c_{n-1}} - t_{c_M-c_{n-1}+1} - \dots - t_{c_M-1} = 0 \end{cases} \quad (1)$$

ただし, $x_0, x_1, \dots, x_{n-1}, t_0, t_1, \dots, t_{c_M-1}$ はバイナリ値である。

$I^{(i)} = \{I_0^{(i)}, I_1^{(i)}, \dots, I_{r_i-1}^{(i)}\}$ を i 行目のインデックスとすると各要素は i 行目の中間変数である。Model 2 よりすべての出力の XOR は次の線形不等式で表すことができる。

$$\begin{cases} t_{I_0^{(0)}} + t_{I_1^{(0)}} + \dots + t_{I_{r_0-1}^{(0)}} - y_0 = 0 \\ t_{I_0^{(1)}} + t_{I_1^{(1)}} + \dots + t_{I_{r_1-1}^{(1)}} - y_1 = 0 \\ \vdots \\ t_{I_0^{(n-1)}} + t_{I_1^{(n-1)}} + \dots + t_{I_{r_{n-1}-1}^{(n-1)}} - y_{n-1} = 0 \end{cases} \quad (2)$$

ただし, $y_0, y_1, \dots, y_{n-1}, t_0, t_1, \dots, t_{c_M-1}$ はバイナリ値である。

式 (1) 及び (2) を合わせたものが M^{PR} の DP 伝搬を表す線形不等式である。

4. Piccolo への応用

4.1 Piccolo の拡散行列 M への応用

Piccolo は 2 節で示したように F 関数内で次の行列 $M_{Piccolo}$ に基づいて S-Box からの出力が拡散され, その出力が再度 S-Box を通過するという構造になっている。

$$M_{Piccolo} = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix}$$

既約多項式は $x^4 + x + 1$ である。ここから $M_{Piccolo}$ のブ
リミティブ表現は以下のように導出できる。

$$M_{Piccolo}^{PR} = \begin{pmatrix} 0 & 1 & 0 & 0 & | & 1 & 1 & 0 & 0 & | & 1 & 0 & 0 & 0 & | & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & | & 0 & 1 & 1 & 0 & | & 0 & 1 & 0 & 0 & | & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & | & 1 & 0 & 1 & 1 & | & 0 & 0 & 1 & 0 & | & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & | & 1 & 0 & 0 & 1 & | & 0 & 0 & 0 & 1 & | & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & | & 0 & 1 & 0 & 0 & | & 1 & 1 & 0 & 0 & | & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & | & 0 & 0 & 1 & 0 & | & 0 & 1 & 1 & 0 & | & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & | & 1 & 0 & 0 & 1 & | & 1 & 0 & 1 & 1 & | & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & | & 1 & 0 & 0 & 0 & | & 1 & 0 & 0 & 1 & | & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & | & 1 & 0 & 0 & 0 & | & 0 & 1 & 0 & 0 & | & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & | & 0 & 1 & 0 & 0 & | & 0 & 0 & 1 & 0 & | & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & | & 0 & 0 & 1 & 0 & | & 1 & 0 & 0 & 1 & | & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & | & 0 & 0 & 0 & 1 & | & 1 & 0 & 0 & 0 & | & 1 & 0 & 0 & 1 \\ \hline 1 & 1 & 0 & 0 & | & 1 & 0 & 0 & 0 & | & 1 & 0 & 0 & 0 & | & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & | & 0 & 1 & 0 & 0 & | & 0 & 1 & 0 & 0 & | & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & | & 0 & 0 & 1 & 0 & | & 0 & 0 & 1 & 0 & | & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & | & 0 & 0 & 0 & 1 & | & 0 & 0 & 0 & 1 & | & 1 & 0 & 0 & 0 \end{pmatrix} \quad (3)$$

$M_{Piccolo}^{PR}$ の入力 DP $D_{\{\vec{x}\}}^{16}$ ($\vec{x} = (x_0, x_1, \dots, x_{15})$) を持
ち、出力は DP $D_{\{\vec{y}\}}^{16}$ ($\vec{y} = (y_0, y_1, \dots, y_{15})$) に準じるとす
る。 $M_{Piccolo}^{PR}$ の非ゼロの要素は 88 個あり、それらの中間
変数 ($t_0 \sim t_{87}$) を以下のように配置する。

$$\begin{pmatrix} 0 & t_7 & 0 & 0 & | & t_{22} & t_{29} & 0 & 0 & | & t_{44} & 0 & 0 & 0 & | & t_{66} & 0 & 0 & 0 \\ 0 & 0 & t_{12} & 0 & | & 0 & t_{30} & t_{34} & 0 & | & 0 & t_{51} & 0 & 0 & | & 0 & t_{73} & 0 & 0 \\ t_0 & 0 & 0 & 0 & | & t_{23} & 0 & t_{35} & t_{39} & | & 0 & 0 & t_{56} & 0 & | & 0 & 0 & t_{78} & 0 \\ t_1 & 0 & 0 & 0 & | & t_{24} & 0 & 0 & t_{40} & | & 0 & 0 & 0 & t_{61} & | & 0 & 0 & 0 & t_{83} \\ \hline t_2 & 0 & 0 & 0 & | & 0 & t_{31} & 0 & 0 & | & t_{45} & t_{52} & 0 & 0 & | & t_{67} & 0 & 0 & 0 \\ 0 & t_8 & 0 & 0 & | & 0 & 0 & t_{36} & 0 & | & 0 & t_{53} & t_{57} & 0 & | & 0 & t_{74} & 0 & 0 \\ 0 & 0 & t_{13} & 0 & | & t_{25} & 0 & 0 & t_{41} & | & t_{46} & 0 & t_{58} & t_{62} & | & 0 & 0 & t_{79} & 0 \\ 0 & 0 & 0 & t_{18} & | & t_{26} & 0 & 0 & 0 & | & t_{47} & 0 & 0 & t_{63} & | & 0 & 0 & 0 & t_{84} \\ \hline t_3 & 0 & 0 & 0 & | & t_{27} & 0 & 0 & 0 & | & 0 & t_{54} & 0 & 0 & | & t_{68} & t_{75} & 0 & 0 \\ 0 & t_9 & 0 & 0 & | & 0 & t_{32} & 0 & 0 & | & 0 & 0 & t_{59} & 0 & | & 0 & t_{76} & t_{80} & 0 \\ 0 & 0 & t_{14} & 0 & | & 0 & 0 & t_{37} & 0 & | & t_{48} & 0 & 0 & t_{64} & | & t_{69} & 0 & t_{81} & t_{85} \\ 0 & 0 & 0 & t_{19} & | & 0 & 0 & 0 & t_{42} & | & t_{49} & 0 & 0 & 0 & | & t_{70} & 0 & 0 & t_{86} \\ \hline t_4 & t_{10} & 0 & 0 & | & t_{28} & 0 & 0 & 0 & | & t_{50} & 0 & 0 & 0 & | & 0 & t_{77} & 0 & 0 \\ 0 & t_{11} & t_{15} & 0 & | & 0 & t_{33} & 0 & 0 & | & 0 & t_{55} & 0 & 0 & | & 0 & 0 & t_{82} & 0 \\ t_5 & 0 & t_{16} & t_{20} & | & 0 & 0 & t_{38} & 0 & | & 0 & 0 & t_{60} & 0 & | & t_{71} & 0 & 0 & t_{87} \\ t_6 & 0 & 0 & t_{21} & | & 0 & 0 & 0 & t_{43} & | & 0 & 0 & 0 & t_{65} & | & t_{72} & 0 & 0 & 0 \end{pmatrix} \quad (4)$$

式 (4) の同じ列にある変数は対応する入力値の *Copy* を
表す線形不等式を構成する変数である。よって $M_{Piccolo}^{PR}$ に
おける *Copy* は次の線形不等式で表される。

$$\begin{cases} x_0 - t_0 - t_1 - t_2 - t_3 - t_4 - t_5 - t_6 = 0 \\ x_1 - t_7 - t_8 - t_9 - t_{10} - t_{11} = 0 \\ x_2 - t_{12} - t_{13} - t_{14} - t_{15} - t_{16} = 0 \\ x_3 - t_{17} - t_{18} - t_{19} - t_{20} - t_{21} = 0 \\ x_4 - t_{22} - t_{23} - t_{24} - t_{25} - t_{26} - t_{27} - t_{28} = 0 \\ x_5 - t_{29} - t_{30} - t_{31} - t_{32} - t_{33} = 0 \\ x_6 - t_{34} - t_{35} - t_{36} - t_{37} - t_{38} = 0 \\ x_7 - t_{39} - t_{40} - t_{41} - t_{42} - t_{43} = 0 \\ x_8 - t_{44} - t_{45} - t_{46} - t_{47} - t_{48} - t_{49} - t_{50} = 0 \\ x_9 - t_{51} - t_{52} - t_{53} - t_{54} - t_{55} = 0 \\ x_{10} - t_{56} - t_{57} - t_{58} - t_{59} - t_{60} = 0 \\ x_{11} - t_{61} - t_{62} - t_{63} - t_{64} - t_{65} = 0 \\ x_{12} - t_{66} - t_{67} - t_{68} - t_{69} - t_{70} - t_{71} - t_{72} = 0 \\ x_{13} - t_{73} - t_{74} - t_{75} - t_{76} - t_{77} = 0 \\ x_{14} - t_{78} - t_{79} - t_{80} - t_{81} - t_{82} = 0 \\ x_{15} - t_{83} - t_{84} - t_{85} - t_{86} - t_{87} = 0 \end{cases} \quad (5)$$

式 (4) の同じ行にある変数は対応する出力値の *XOR* を
表す線形不等式を構成する変数である。よって $M_{Piccolo}^{PR}$ に
おける *XOR* は次の線形不等式で表される。

$$\begin{cases} t_7 + t_{22} + t_{29} + t_{44} + t_{66} - y_0 = 0 \\ t_{12} + t_{30} + t_{34} + t_{51} + t_{73} - y_1 = 0 \\ t_0 + t_{17} + t_{23} + t_{35} + t_{39} + t_{56} + t_{78} - y_2 = 0 \\ t_1 + t_{24} + t_{40} + t_{61} + t_{83} - y_3 = 0 \\ t_2 + t_{31} + t_{45} + t_{52} + t_{67} - y_4 = 0 \\ t_8 + t_{36} + t_{53} + t_{57} + t_{74} - y_5 = 0 \\ t_{13} + t_{25} + t_{41} + t_{46} + t_{58} + t_{62} + t_{79} - y_6 = 0 \\ t_{18} + t_{26} + t_{47} + t_{63} + t_{84} - y_7 = 0 \\ t_3 + t_{27} + t_{54} + t_{68} + t_{75} - y_8 = 0 \\ t_9 + t_{32} + t_{59} + t_{76} + t_{80} - y_9 = 0 \\ t_{14} + t_{37} + t_{48} + t_{64} + t_{69} + t_{81} + t_{85} - y_{10} = 0 \\ t_{19} + t_{42} + t_{49} + t_{70} + t_{86} - y_{11} = 0 \\ t_4 - t_{10} - t_{28} - t_{50} - t_{77} - y_{12} = 0 \\ t_{11} + t_{15} + t_{33} + t_{55} + t_{82} - y_{13} = 0 \\ t_5 + t_{16} + t_{20} + t_{38} + t_{60} + t_{71} + t_{87} - y_{14} = 0 \\ t_6 + t_{21} + t_{43} + t_{65} + t_{72} - y_{15} = 0 \end{cases} \quad (6)$$

以上より、 $M_{Piccolo}^{PR}$ を表す線形不等式は式 (5) 及び (6)
で構成される。

4.2 Piccolo に対する MILP 手法の適用

本節では Ling らの MILP 手法を Piccolo へ適用するに
あたり適用要領を示す。図 5 は Piccolo の構造に対して前
節で述べた MILP 手法を適用するにあたり図 1 のラウン
ド関数内を具体化したものである。Piccolo の MILP 手法
による DP 探索アルゴリズムの実装において、伝搬規則を

表すモデルを図5①～⑤に示す箇所に適用した．それぞれ①Copy, ②S-Box, ③ $M_{Piccolo}^{PR}$, ④S-Box, ⑤XORの伝搬規則を表している．図5において左側のみ伝搬規則箇所を示しているが右側も同様である．またRPについてはXORを表す線形不等式の出力をRPにより変化された後の変数にすることにより適用する．

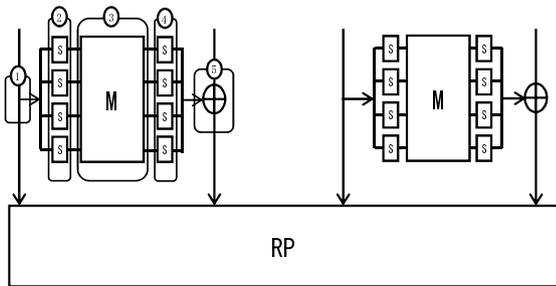


図5 MILP手法の適用

4.3 MILP手法によるDivision Propertyの探索結果

本研究では久保らの結果を検証するため，文献[6]と同様に8, 12, 24, 32, 48, 63次のDP探索をMILP手法で実行した．この結果を表2に示す．

表2 MILP手法による探索結果

次数	発見した段数	探索に要した時間 [s]
8	4	34.73 (6.00)
12	4	110.70 (9.87)
24	5	10.68 (19.07)
32	6	26.40 (27.96)
48	6	86894.30 (45.55)
63	7	1186.53 (100.78)

探索に要した時間は，探索が完了するまでの時間を表す．(カッコ)内の数値は発見した段数より1段多い段数に対して探索を実行し，終了するまでに経過した時間を示している．なお，久保らは48次まで網羅的に探索したと述べているが要した探索時間は記述されていない．そのためMILP手法の高速性に関して簡単な比較はできないが，MILPは充分実用的にDP探索が可能であると結論できる．表3に久保らの結果との比較を示す．

8次，12次，48次においては久保らの結果と同様の結果が得られた．一方で表3に太字で示すように24次，32次，63次においては久保らの結果より1段多い結果を得ることがわかった．久保らにより24次，32次のDP探索の結果については既知の高階差分特性より1段劣化していることが示されている．本研究ではMILP手法を用いたDP探索は24次，32次のDP探索における特性の劣化を防止

表3 久保らの結果との比較

次数	MILP	久保らの結果
8	4	4
12	4	4
24	5	4
32	6	5
48	6	6
63	7	6

することが可能であるということがわかった．

5. 考察

本研究において上記に示した通り24次，32次においてはDP探索における劣化を防止することが可能であった．また，63次においては久保らの結果より1段多い7段特性を発見することができた．本節ではなぜDP探索における特性の劣化を防止することができたのか，なぜ63次においては1段多い特性を発見することができたのかを考察する．

まず，久保らにより[6]においてDPの劣化に関する検証が行われている．検証の結果，Copyによる特性を打ち消す \vec{k} の生成及びSizeReduceにより特性が劣化することが報告されている．藤堂らの手法によるFeistel構造のDP探索については，DPを探索する際にSizeReduceを使用している．これは $W(\vec{k}) \supseteq W(\vec{k}')$ を満足する $\vec{k} \in \mathbb{K}'$ と $\vec{k}' \in \mathbb{K}$ が存在するとき， \vec{k} を \mathbb{K} を削除する関数である．これを使用して藤堂らの手法は \mathbb{K} を収束させていきDP特性の探索効率を向上させている．一方，MILP手法については探索する段数及び次数は手動で設定する必要がある．しかし，特性の劣化を引き起こすと考えられているSizeReduceを使用せず各段においてDP伝搬特性を表す線形不等式を生成し，MILPオプティマイザーを利用して最適化させる．そのため特性の劣化は起こらず正確な探索が実行できる．

また，63次においては文献[6]において久保らにより一部においてCopy及びSizeReduceによる劣化が確認できると示されている．この要因についても先に述べた通り，藤堂らの手法によるDP探索を行っているためであると考えられる．しかし，本研究で使用したMILP手法では，先に述べた通りSizeReduceを使用せずに探索を行っている．そのため63次においてもSizeReduceを使用することによる特性の劣化は起こらなかった．よって，正確な探索が可能となり久保らの結果よりも1段多い7段目特性を発見できたと考える．

6. 結論

本研究では，Lingらの手法を用いてMILP手法でPiccoloのDP特性の探索を行った．検証実験の結果，SizeReduceを使用しないMILP手法においては劣化が見られず既知

の高階差分特性と同様の結果が得られた。そのため、藤堂の手法を用いた DP 探索では SizeReduce を使用しているために特性の劣化を引き起こしているという久保らの劣化に関する検証の正当性を確認できた。また、DP 探索において劣化を引き起こさず正確な探索が可能となるという MILP の有用性についても確認することができた。

また、久保らは文献 [6] において、63 次においても劣化が確認できるため高階差分の調査では 1 段伸びた 7 段特性が現れると推測している。この推測について、MILP 手法において SizeReduce による劣化のない正確な DP 探索を行った結果として 7 段特性が発見できたため妥当であると考えられる。

さらに、MILP 手法は Piccolo の既知の高階差分特性とほぼ同等の結果を得ることが可能であり、既存の高階差分特性の調査では行われていなかった次数においても適用できるため MILP 手法は Integral distinguisher を探索する有効な手段であると考えられる。

本研究においては、SizeReduce を使用しない MILP 手法を用いて DP 探索を行い、24, 32 次, 63 次においては藤堂らの手法による DP 探索の劣化を回避して正確な探索を行うことができた。しかし、48 次元においては既知の高階差分特性と比べて劣化が見られる結果となった。この劣化の原因として考えられるのは *Copy* による特性を打ち消すモデルの生成であると考えられる。このため、*Copy* による劣化を防止する手法について検討する。

参考文献

- [1] Y. Todo: Structural Evaluation by Generalized Integral Property, EUROCRYPT 2015, LNCS, vol 9056, part 1, pp.287-314(2015).
- [2] Y. Todo: Integral Cryptanalysis on Full MISTY1, CRYPTO 2015, LNCS, vol 9215, pp.413-432(2015).
- [3] Z. Xiang, W. Zhang, Z. Bao, and D. Lin: Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers, ASIACRYPT 2016, Proceedings, Part I, pp.648-678(2016).
- [4] L. Sun, W. Wang, and M. Wang: MILP-Aided Bit-Based Division Property for Primitives with Non-Bit-Permutation Linear Layers, IACR Cryptology ePrint Archive(2016).
- [5] K. Shibutani, T. Isobe, H. Hawatari, A. Mutuda, T. Akishita, and T. Shirai: Piccolo: An Ultra-Lightweight Blockcipher, CHES2011, LNCS, vol 6917, pp.342-357(2011).
- [6] 久保卓也, 五十嵐保隆, 金子敏信: 軽量ブロック暗号 Piccolo の Division Property 及び高階差分特性の比較・検証, 研究報告セキュリティ心理学とトラスト (SPT), 2016-SPT-17 巻, 27 号, pp.1-6(2016).
- [7] <http://www.gurobi.com/>
- [8] <http://www.sagemath.org/>
- [9] S. Sun, L. Hu, M. Wang, P. Wang, K. Qiao, X. Ma, D. Shi, L. Song, and K. Fu, Towards finding the best characteristics of some bit-oriented block ciphers and automatic enumeration of (related-key) differential and linear characteristics with predefined properties, Technical report, Cryptology ePrint Archive, Report 2014/747(2014).