An investigation on trail search algorithms of block ciphers

A Project Report Submitted by

Arindam Ghosh

under guidance of

Dr. Bimal Mandal

in partial fulfillment of the requirements for the award of the degree of

Master in Science in Mathematics

Indian Institute of Technology Jodhpur

Mathematics

April, 2025

Declaration

I hereby declare that the work presented in this Project Report titled **An investigation on trail search algorithms of block ciphers** submitted to the Indian Institute of Technology Jodhpur in partial fulfillment of the requirements for the award of the degree of Master in Science in Mathematics, is a genuine record of the research conducted under the guidance of **Bimal Mandal**. The contents of this Project Report, in whole or in part, have not been submitted to and will not be submitted by me to any other Institute or University for the purpose of obtaining any degree or diploma.

Signature

Arindam Ghosh

m23ma1004

Certificate

This is to certify that the Project Report titled **"An investigation on trail search algorithms of block ciphers"**, submitted by **Arindam Ghosh (m23ma1004)** to the Indian Institute of Technology Jodhpur for the award of the degree of Master in Science in Mathematics, is a genuine record of the research work conducted by him under my supervision. To the best of my knowledge, the contents of this report, in whole or in part, have not been submitted to any other Institute or University for the purpose of obtaining any degree or diploma.

Signature

Bimal Mandal

Acknowledgements

I would like to express my sincere gratitude to all those who contributed to the successful completion of this project. First and foremost, I extend my heartfelt thanks to my supervisor, **Dr. Bimal Mandal**, for his guidance, support, and invaluable insights throughout the research process. Their expertise and encouragement were instrumental in shaping my understanding of the topic.

I would also like to thank **Mr. Ranit Dutta** and my peers for their constructive feedback and collaborative spirit, which enriched my learning experience. Finally, I would like to acknowledge the unwavering support of my family and friends, whose encouragement and understanding have been a constant source of motivation throughout this journey.

Thank you all for your contributions and support.

Abstract

This project investigates trail search algorithms for block ciphers, with a focus on identifying optimal differential trails to enhance cryptographic security. Using the lightweight MIDORI cipher as a case study, the research analyzes AES-like structures and implements two key algorithms: the Difference Distribution Table method and Matsui's depth-first search technique. Also this report investigates permutation characteristics in AES-like block ciphers, with a focus on how linear layers composed of permutation matrices influence the structure and equivalence of differential trails. We define and analyze the concept of permutation characteristics, leveraging their properties to reduce redundancy in trail search algorithms. By identifying equivalence classes via first-round word-wise permutations, we construct reduced activity patterns that significantly reduces the search space. These methods provide computational tools to evaluate the resistance of block ciphers to differential cryptanalysis. The study highlights opportunities to refine these algorithms and adapt them to various cipher designs, contributing to the development of more secure encryption systems.

Contents

1 Introduction

Currently, differential cryptanalysis (DC) [4] and linear cryptanalysis (LC) [1] are the most essential attacks for block ciphers. In fact, resistance against DC and LC is considered as the most important factor to determine the number of rounds for an iterative block cipher.

Both DC and LC begin by building a DC/LC distinguisher with high probability, as this probability largely influences the complexity of the attack and its success rate. Differential and linear distinguishers are obtained from differential and linear trails (also known as characteristics).

Evaluating the upper bound for the probability of trails can be done by two primary methods: one approach involves identifying the minimum number of active S-boxes across all possible trails, while the other involves exhaustively searching all concrete trails to find the best trails with the highest probability. The first approach utilizes the properties of S-boxes, mixed integer linear programming (MILP)-aided search, the wide trail strategy to find meaningful bounds, which are often applied in the design of various block cipher proposals. However, this approach only yields an upper bound for the probability, without identifying any best trails, so it may not ensure a precise or tight upper bound. Therefore, identifying the best trail remains a primary concern not only for carrying out attacks but also for designing secure block ciphers.

At EUROCRYPT'94 [5], Matsui introduced a dedicated search algorithm to find the best differential and linear trails. This algorithm was later refined by introducing a pre-computation phase to discard search patterns for each round if certain conditions were not met.

2 Preliminaries

2.1 AES-like ciphers

An AES-like cipher[3] is a key-alternating iterated block cipher which applies an AES-like round function **R** in all rounds. Thus, an AES-like cipher consisting of R-rounds is defined as

$$E^{(R)} = \bigoplus_{k_R} \circ \mathrm{R} \circ \cdots \circ \bigoplus_{k_2} \circ \mathrm{R} \circ \bigoplus_{k_1} \circ \mathrm{R} \circ \bigoplus_{k_1}$$

where $\bigoplus_{k_i}(x) = x \oplus k_i$ is round key addition.

Definition 1. An AES-like round function[3] is $\mathrm{R} : \mathbb{F}_2^{wmn} \to \mathbb{F}_2^{wmn}$ where $m \times n$ is the state dimension and w is word size. R is composed of an non-linear Substitution-layer Sub and a Linear-layer Lin. $R = Lin \circ Sub$.

Block/State of AES-like cipher

$$\mathbb{X} = \begin{bmatrix} \mathbb{X}[0] & \mathbb{X}[m] & \mathbb{X}[2m] & \cdots & \mathbb{X}[(n-1)m] \\ \mathbb{X}[1] & \mathbb{X}[m+1] & \mathbb{X}[2m+1] & \cdots & \mathbb{X}[(n-1)m+1] \\ \mathbb{X}[2] & \mathbb{X}[m+2] & \mathbb{X}[2m+2] & \cdots & \mathbb{X}[(n-1)m+2] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbb{X}[m-1] & \mathbb{X}[2m-1] & \mathbb{X}[3m-1] & \cdots & \mathbb{X}[nm-1] \end{bmatrix}$$

$$= \begin{bmatrix} \mathbb{X}\langle 0\rangle & \mathbb{X}\langle 1\rangle & \mathbb{X}\langle 2\rangle & \cdots & \mathbb{X}\langle n-1\rangle \end{bmatrix}$$

where $\mathbb{X} \in \mathbb{F}_2^{wmn}$, $\mathbb{X}[i] \in \mathbb{F}_2^w$, $\mathbb{X}\langle k\rangle \in \mathbb{F}_2^{wm}$ and $w =$ No. of bits in each word, $m =$ No of rows in the State, $n =$ No. of columns in the State.

Substitution

Let $\mathbb{X} \in \mathbb{F}_2^{wmn}$ be an input Block. The **Substitution** operation, denoted as

$$\text{Sub} : \mathbb{F}_2^{wmn} \to \mathbb{F}_2^{wmn},$$

is defined using an S-box $S : \mathbb{F}_2^w \to \mathbb{F}_2^w$. The output $\mathbb{Y} = \text{Sub}(\mathbb{X})$ is computed word-wise as:

$$\mathbb{B}[i] = S(\mathbb{X}[i]) \quad \text{for all } i = 0, 1, 2, \ldots, mn-1.$$

Types of Linear Layer

Linear layers in block ciphers can be broadly classified into two categories:

- **Bit-based Linear Layer**
- **Non-bit-based Linear Layer**

Bit-based Linear Layer: A **bit-based linear layer** operates directly on individual bits of the cipher state. Let the state be represented as

$$\mathbb{Y} = \mathbb{Y}_0 \parallel \mathbb{Y}_1 \parallel \cdots \parallel \mathbb{Y}_{wmn-1}, \quad \mathbb{Y}_i \in \mathbb{F}_2, \text{ i-th bit of state } \mathbb{Y}.$$

The layer applies a fixed permutation $\sigma \in S_{wmn}$ to reorder the bits as:

$$\text{Lin} : \mathbb{F}_2^{wmn} \to \mathbb{F}_2^{wmn},$$

$$\mathbb{X}' = \text{Lin}(\mathbb{Y}),$$

$$\mathbb{X}'_i = \mathbb{Y}_{\sigma(i)} \quad \text{for all } i \in \{0, 1, \ldots, wmn-1\}.$$

Equivalently, the output can be expressed as:

$$\mathrm{Lin}(\mathbb{Y}) = \mathbb{Y}_{\sigma(0)} \parallel \mathbb{Y}_{\sigma(1)} \parallel \cdots \parallel \mathbb{Y}_{\sigma(wmn-1)}.$$

Non-bit-based Linear Layer: A **non-bit-based linear layer** operates on larger sub-blocks or words. It typically consists of the following two components:

- **MixColumn:** A linear transformation applied column-wise via matrix multiplication.
- **Shuffle:** A word-level permutation defined by $\sigma \in S_{mn}$, where S_{mn} is the symmetric group over mn words.

Based on the order in which these two operations are applied, there are two common architectural variants:

- **M-S (Mix-then-Shuffle):** Apply MixColumn first, followed by Shuffle.
- **S-M (Shuffle-then-Mix):** Apply Shuffle first, followed by MixColumn.

MixColumn and Shuffle

The **MixColumn** operation $\mathrm{Mix} : \mathbb{F}_2^{wmn} \to \mathbb{F}_2^{wmn}$ uses a linear transformation $\mathrm{M} : \mathbb{F}_2^{wm} \to \mathbb{F}_2^{wm}$ column-wise.
$\mathbb{Z} = \mathrm{Mix}(\mathbb{Y})$ is defined as

$$\mathbb{Z}\langle k \rangle = \mathrm{M}(\mathbb{Y}\langle k \rangle) \quad \forall k = 0, 1, \ldots, n-1,$$

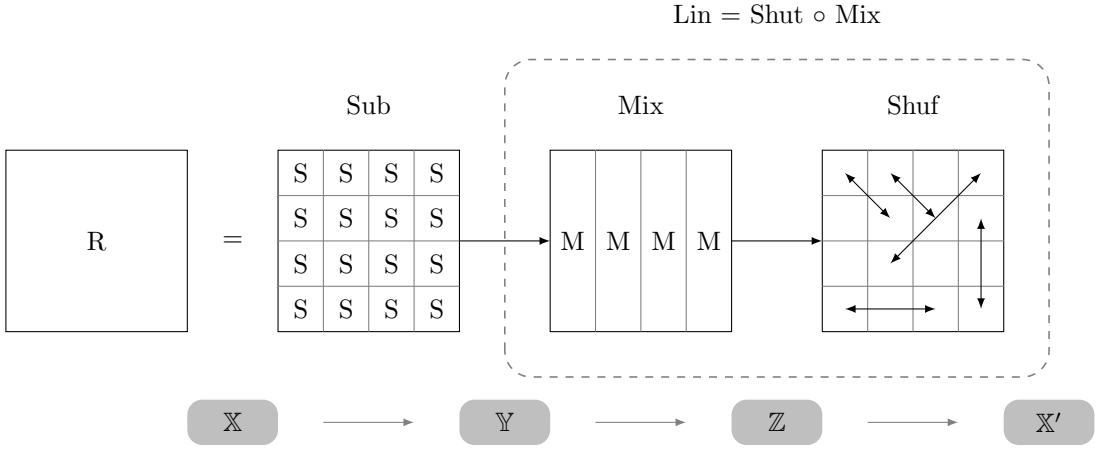
where $\mathbb{Y}\langle k \rangle \in \mathbb{F}_2^{wm}$ denotes the k-th column.

The **Shuffle** operation reorders the words in the state using a fixed permutation $\sigma \in S_{mn}$.
$\mathrm{Shuf} : \mathbb{F}_2^{wmn} \to \mathbb{F}_2^{wmn}$ is defined as

$$\mathbb{X}' = \mathrm{Shuf}(\mathbb{Z}),$$

$$\mathbb{X}'[i] = \mathbb{Z}[\sigma(i)] \quad \text{for all } i = 0, 1, \ldots, mn-1,$$

where $\sigma \in S_{mn}$ is a permutation over word indices.



Example 1. **MIDORI** [6] is a family of 2 AES-like block ciphers: **MIDORI-64** and **MIDORI-128**. Both ciphers use

$$\sigma_{\mathbf{MIDORI}} = (1\ 7\ 12\ 10)(2\ 14\ 4\ 5)(3\ 9\ 8\ 15)(6\ 11) \in \mathrm{S}_{16}$$

and

$$M_{\mathbf{MIDORI}} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

For **MIDORI-64** $w = 4$ and $m = n = 4$, Sub is defined using one S-box $Sb_0 : \mathbb{F}_2^4 \to \mathbb{F}_2^4$ by the lookup table

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$Sb_0(x)$	C	A	D	3	E	B	F	7	8	9	1	5	0	2	4	6

For **MIDORI-128** $w = 8$ and $m = n = 4$, Sub is defined using 4 different S-boxs $SSb_i : \mathbb{F}_2^8 \to \mathbb{F}_2^8$, $i \in \{0, 1, 2, 3\}$, which are derived from 1 S-box $Sb_1 : \mathbb{F}_2^4 \to \mathbb{F}_2^4$ and 4 Permutations $p_i \in \mathrm{S}_8;\ i \in \{0, 1, 2, 3\}$ as

$$SSb_i = p_i^{-1} \circ (Sb_1 \parallel Sb_1) \circ p_i, \qquad \forall i \in \{0, 1, 2, 3\}$$

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$Sb_1(x)$	C	A	D	3	E	B	F	7	8	9	1	5	0	2	4	6

$$p_0 = (0\ 4)(2\ 6) \qquad p_1 = (0\ 1\ 6\ 3)(2\ 7\ 4\ 5)$$
$$p_2 = (0\ 2\ 4\ 6)(1\ 3)(5\ 7) \qquad p_3 = (0\ 7\ 6\ 5)(1\ 4\ 3\ 2)$$

2.2 Differential probability and weight

Definition 2. Let $g : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a function. A difference propagation $\nabla \xrightarrow{g} \nabla'$ has a differential probability defined as

$$\Pr(\nabla \xrightarrow{g} \nabla') = \frac{|\{x : g(x \oplus \nabla) = g(x) \oplus \nabla'\}|}{2^n}.$$

Definition 3. The weight of a difference propagation $\nabla \xrightarrow{g} \nabla'$ is defined as

$$\mathrm{W}(\nabla \xrightarrow{g} \nabla') = -\log_2 \Pr(\nabla \xrightarrow{g} \nabla').$$

Also the maximum and minimum differential weights of g are defined as

$$\overline{\mathrm{W}}_g = max\{\mathrm{W}(\nabla \xrightarrow{g} \nabla') : \nabla \neq 0 \ , \ \Pr[\nabla \xrightarrow{g} \nabla'] \neq 0\}$$

$$\underline{\mathrm{W}}_g = min\{\mathrm{W}(\nabla \xrightarrow{g} \nabla') : \nabla \neq 0\}$$

Definition 4. A **R-round differential trail** over an AES-like cipher is a sequence of difference propagations over each round. Let $\mathbb{T}$ be a R-round trail, then $\mathbb{T} = [\ \mathbb{T}_1, \mathbb{T}_2, \ldots, \mathbb{T}_{\mathrm{R}}\ ]$ where $\mathbb{T}_i$ is difference propagation in the i-th round.

$$\cdots \quad \mathbb{T}_{i-1}(\mathbb{X}') = \mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{Sub}} \mathbb{T}_i(\mathbb{Y}) \xrightarrow{\text{Mix}} \mathbb{T}_i(\mathbb{Z}) \xrightarrow{\text{Shuf}} \mathbb{T}_i(\mathbb{X}') = \mathbb{T}_{i+1}(\mathbb{X}) \quad \cdots$$

Notations for Trail

$\mathbb{T}$	Differential Trail
$\mathbb{T}_i(\mathbb{X})$	Input difference of **Sub** in the i-th round
$\mathbb{T}_i(\mathbb{Y})$	Output difference of **Sub** in the i-th round also input difference of **Mix** in the i-th round
$\mathbb{T}_i(\mathbb{Z})$	Output difference of **Mix** in the i-th round also input difference of **Shuf** in the i-th round
$\mathbb{T}_i(\mathbb{X}')$	Output difference of **Shuf** in the i-th round
$\mathbb{T}_i(\mathbb{W})$	Weight of the difference propagation $\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{Sub}} \mathbb{T}_i(\mathbb{Y})$
$\mathbb{T}_i(\mathbb{W}[j])$	Weight of the difference propagation $\mathbb{T}_i(\mathbb{X}[j]) \xrightarrow{\text{S}} \mathbb{T}_i(\mathbb{B}[j])$

Since round key additions are considered as identities in differential trails, the output differences of one round's propagation always equal to the input differences of the next round's propagation, i.e. $\mathbb{T}_i(\mathbb{X}') = \mathbb{T}_{i+1}(\mathbb{X})$

Definition 5. Expected probability of a differential trail $\mathbb{T}$ is defined as

$$\mathrm{EDP}(\mathbb{T}) = \prod_{i=1}^{R} \Pr\left[\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{R}} \mathbb{T}_i(\mathbb{X}')\right]$$

As R = Lin $\circ$ Sub, the probability of difference propagation across each round can be broken down into

the probabilities associated with the Sub and Lin as

$$\Pr\left[\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{R}} \mathbb{T}_i(\mathbb{X}')\right] = \Pr\left[\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{Sub}} \mathbb{T}_i(\mathbb{Y})\right] \times \Pr\left[\mathbb{T}_i(\mathbb{Y}) \xrightarrow{\text{Lin}} \mathbb{T}_i(\mathbb{X}')\right]$$

Also,

$$\Pr\left[\mathbb{T}_i(\mathbb{Y}) \xrightarrow{\text{Lin}} \mathbb{T}_i(\mathbb{X}')\right] = \begin{cases} 1 & \text{if } \mathbb{T}_i(\mathbb{X}') = \text{Lin}(\mathbb{T}_i(\mathbb{Y})), \\ 0 & \text{otherwise} \end{cases}$$

Therefore the difference propagation $\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{Sub}} \mathbb{T}_i(\mathbb{Y})$ determines $\Pr\left[\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{R}} \mathbb{T}_i(\mathbb{X}')\right]$ as long as $\mathbb{T}_i(\mathbb{X}') = \text{Lin}(\mathbb{T}_i(\mathbb{Y}))$. Further $\Pr\left[\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{Sub}} \mathbb{T}_i(\mathbb{Y})\right]$ can be derived from the probabilities of difference propagations over mn S-boxes as

$$\Pr\left[\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{Sub}} \mathbb{T}_i(\mathbb{Y})\right] = \prod_{j=0}^{mn-1} \Pr\left[\mathbb{T}_i(\mathbb{X})[j] \xrightarrow{\text{S}} \mathbb{T}_i(\mathbb{Y})[j]\right]$$

Combining all the above results and assuming that $\mathbb{T}_i(\mathbb{X}') = \text{Lin}(\mathbb{T}_i(\mathbb{Y})) \ \ \forall i$, we get

$$\begin{aligned} \text{EDP}(\mathbb{T}) &= \prod_{i=1}^{R} \Pr\left[\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{R}} \mathbb{T}_i(\mathbb{X}')\right] \\ &= \prod_{i=1}^{R} \Pr\left[\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{Sub}} \mathbb{T}_i(\mathbb{Y})\right] \times \Pr\left[\mathbb{T}_i(\mathbb{Y}) \xrightarrow{\text{Lin}} \mathbb{T}_i(\mathbb{X}')\right] \\ &= \prod_{i=1}^{R} \Pr\left[\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{Sub}} \mathbb{T}_i(\mathbb{Y})\right] \\ &= \prod_{i=1}^{R} \prod_{j=0}^{mn-1} \Pr\left[\mathbb{T}_i(\mathbb{X}[j]) \xrightarrow{\text{S}} \mathbb{T}_i(\mathbb{B}[j])\right] \end{aligned}$$

Definition 6. The weight of an R-round differential trail $\mathbb{T}$ is defined as

$$\begin{aligned} \text{W}(\mathbb{T}) &= -\log_2 \text{EDP}(\mathbb{T}) \\ &= -\log_2 \left(\prod_{i=1}^{R} \Pr\left[\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{Sub}} \mathbb{T}_i(\mathbb{Y})\right] \right) = \sum_{i=1}^{R} -\log_2 \left(\Pr\left[\mathbb{T}_i(\mathbb{X}) \xrightarrow{\text{Sub}} \mathbb{T}_i(\mathbb{Y})\right] \right) = \sum_{i=1}^{R} \mathbb{T}_i(\mathbb{W}) \\ \text{W}(\mathbb{T}) &= -\log_2 \left(\prod_{i=1}^{R} \prod_{j=0}^{mn-1} \Pr\left[\mathbb{T}_i(\mathbb{X})[j] \xrightarrow{\text{S}} \mathbb{T}_i(\mathbb{Y})[j]\right] \right) \\ &= \sum_{i=1}^{R} \sum_{j=0}^{mn-1} -\log_2 \left(\Pr\left[\mathbb{T}_i(\mathbb{X})[j] \xrightarrow{\text{S}} \mathbb{T}_i(\mathbb{Y})[j]\right] \right) = \sum_{i=1}^{R} \sum_{j=0}^{mn-1} \mathbb{T}_i(\mathbb{W}[j]) \end{aligned}$$

Algorithm 1 Constructing a Difference Distribution Table (DDT)

Input: S-box S of size n, where $S : \{0, 1, \ldots, 2^n - 1\} \to \{0, 1, \ldots, 2^n - 1\}$
Output: Difference Distribution Table $DDT[\Delta X][\Delta Y]$

Initialize $DDT[\Delta X][\Delta Y] = 0$ for all possible input and output differences $\Delta X, \Delta Y$
for each $\Delta X \in \mathbb{F}_2^n$ **do**
    for each $X \in \mathbb{F}_2^n$ **do**
        $X' \leftarrow X \oplus \Delta X$
        Compute output differences: $\Delta Y \leftarrow S(X) \oplus S(X')$
        Increment $DDT[\Delta X][\Delta Y]$ by 1
    end for
end for
return The completed Difference Distribution Table DDT

3 Trail Search Algorithm using DD Table

Using the DD Table, we can find probability of difference propagation through S as

$$\Pr[\Delta X \xrightarrow{\mathrm{S}} \Delta Y] = \frac{\mathrm{DDT}[\Delta X][\Delta Y]}{2^n}$$

Algorithm 2 Finding Best Trail and Probability using DD Table

Input: Difference Distribution Table $\mathrm{DDT}[\alpha][\beta]$, Number of Rounds R
Output: Best Trail and its Probability

Initialize best probability $P_{best} = 0$
Initialize best trail $Trail_{best} = \emptyset$
for each initial input difference $\alpha_0 \in \mathbb{F}_2^n$ **do**
    Initialize trail probability $P_{total} = 1$
    Initialize current trail $Trail = \emptyset$
    for each round $i \in \{0, 1, 2, \ldots, R-1\}$ **do**
        Find $\beta_i = \mathrm{argmax}_{\beta \in \mathbb{F}_2^n} \mathrm{DDT}[\alpha_i][\beta]$
        Compute $\alpha_{i+1} = \mathrm{Lin}(\beta_i)$
        Update trail probability $P_{total} = P_{total} \cdot \Pr[\alpha_i \xrightarrow{Sub} \beta_i]$
        Append (α_i, β_i) to $Trail$
    end for
    if $P_{total} > P_{best}$ **then**
        Update best probability $P_{best} = P_{total}$
        Update best trail $Trail_{best} = Trail$
    end if
end for
return $Trail_{best}$ and P_{best}

Example 2. Here is the DD Table for Sb_0 S-box from **MIDORI-64** cipher.

ΔY \ ΔX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1	00	02	04	00	02	02	02	00	02	00	00	00	00	00	02	00
2	00	04	00	00	04	00	00	00	00	04	00	00	04	00	00	00
3	00	00	00	00	02	00	04	02	02	02	00	00	00	02	00	02
4	00	02	04	02	02	02	00	00	02	00	00	02	00	00	00	00
5	00	02	00	00	02	00	00	04	00	02	04	00	02	00	00	00
6	00	02	00	04	00	00	00	02	02	00	00	00	02	02	00	02
7	00	00	00	02	00	04	02	00	00	00	00	02	00	04	02	00
8	00	02	00	02	02	00	02	00	00	02	00	02	02	00	02	00
9	00	00	04	02	00	02	00	00	02	02	00	02	02	00	00	00
A	00	00	00	00	00	04	00	00	00	00	04	00	00	04	00	04
B	00	00	00	00	02	00	00	02	02	02	00	04	00	02	00	02
C	00	00	04	00	00	02	02	00	02	02	00	00	02	00	02	00
D	00	00	00	02	00	00	02	04	00	00	04	02	00	00	02	00
E	00	02	00	00	00	00	00	02	02	00	00	00	02	02	04	02
F	00	00	00	02	00	00	02	00	00	00	04	02	00	00	02	04

This table is constructed by the above algorithm on a dedicated implementation of **MIDORI-64** cipher in C++.

This search algorithm can be classified as a type of **Greedy Search** algorithm. It starts with a initial point and proceed forward by making the locally optimal choice at each step, hoping these locally optimal choices will lead to a globally optimal solution. But it is not guaranteed to reach a global optimal always.

3.1 Modifying goal using weight

So far our goal is to find trail with best (highest) probability, which always lie in $[0, 1]$. In the table below, we showed a simple relation between probability and weight of a trail.

Probability of trail	Weight of trail
0	∞
1/4	2
1/2	1
1	0

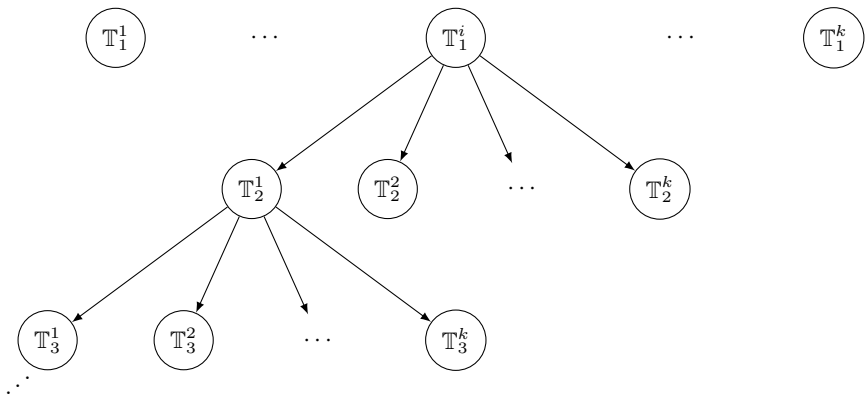
So now our modified goal is to find trail with best (least) weight.

3.2 Why weight instead of probability

- Weight transforms small probabilities into more manageable values.
- Multiplication of probabilities transforms into addition of weights, generally multiplication requires more logic gates than addition, using weights is more efficient than probability.

4 Matsui's Search Algorithm ($\mathcal{M}$) [1]

This algorithm $\mathcal{M}$ focuses on finding the best differential trails by utilizing a **depth-first search (DFS)** approach combined with a **branch-and-bound** technique. $\mathcal{M}$ needs the best weights for rounds 1 through $R-1$, denoted as $\mathbb{B}[1,\ldots,R-1]$, as input. The 1-round best weight $\mathbb{B}[1]$ of an AES-like cipher can be intuitively evaluated as the minimum weight of S-box $\underline{W}$, and $\mathbb{B}[2,\ldots,R-1]$ can be obtained by applying $\mathcal{M}$ inductively from 2 to $R-1$ rounds.



Trail search tree

Since the weight or probability of a round trail depends only on the difference propagation through the non-linear **Sub** layer, the main components of each round trail $\mathbb{T}_i$ are $\mathbb{T}_i.\mathbb{X}$ and $\mathbb{T}_i.\mathbb{Y}$, both in $\mathbb{F}_2^{wmn}$.

Therefore, the number of possible choices for $\mathbb{T}_i$ is:

$$2^{wmn} \times 2^{wmn} = 2^{2wmn} = 4^{wmn} = k$$

4.1 Pruning Condition by Matsui [1]

Proposition 1. Let $R \geq 1$ and $\mathbb{T}$ be an R-round non-trivial trail. For any $1 \leq r \leq R$ and $0 \leq c < mn$, if $W(\mathbb{T}) \leq \overline{B}$, then

$$\sum_{i=1}^{r-1} \mathbb{T}_i(\mathbb{W}) + \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W})[j] + \mathbb{B}[R-r] \leq \overline{B}$$

where:

- R is the number of rounds.
- $W(\mathbb{T})$ is the weight of the trail $\mathbb{T}$.
- $\overline{B}$ is an upper bound of trail weight.
- $\mathbb{B}[i]$ is the best i-round trail weight. For consistency, $\mathbb{B}[0] = 0$.

Proof

We start with the definition of trail weight:

$$W(\mathbb{T}) = \sum_{i=0}^{R} \mathbb{T}_i(\mathbb{W}) \quad \text{and} \quad W(\mathbb{T}) \leq \overline{B}$$

and

$$\mathbb{B}[i] = \text{best (minimum) weight of an } i\text{-round trail}$$

then

$$\Rightarrow \quad \mathbb{B}[R-r] \leq \underbrace{\sum_{i=r+1}^{R} \mathbb{T}_i(\mathbb{W})}_{\text{weight of last } (R-r) \text{ rounds}}$$

$$\therefore \quad \overline{B} \geq \sum_{i=0}^{R} \mathbb{T}_i(\mathbb{W}) = \sum_{i=0}^{r} \mathbb{T}_i(\mathbb{W}) + \sum_{i=r+1}^{R} \mathbb{T}_i(\mathbb{W})$$

$$\geq \sum_{i=0}^{r} \mathbb{T}_i(\mathbb{W}) + \mathbb{B}[R-r]$$

$$= \sum_{i=0}^{r-1} \mathbb{T}_i(\mathbb{W}) + \mathbb{T}_r(\mathbb{W}) + \mathbb{B}[R-r]$$

Since:

$$\mathbb{T}_r(\mathbb{W}) = \sum_{j=0}^{mn-1} \mathbb{T}_r(\mathbb{W})[j] \geq \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W})[j]$$

$$\Longrightarrow \quad \overline{B} \geq \sum_{i=0}^{r-1} \mathbb{T}_i(\mathbb{W}) + \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W})[j] + \mathbb{B}[R-r] \qquad [Proved]$$

This pruning condition is checked at each round. If the pruning condition fails, i.e.,

$$\overline{B} < \sum_{i=0}^{r-1} \mathbb{T}_i(\mathbb{W}) + \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W})[j] + \mathbb{B}[R-r]$$

then

$$\overline{B} < \mathbb{W}(\mathbb{T})$$

which implies that the weight of the current trail exceeds the upper bound $\overline{B}$. In such a case, the algorithm discards this trail and proceeds to the next possible candidate.

4.2 Structure of Matsui's Search Algorithm

The algorithm $\mathcal{M}$ is divided into four procedures: **EstimateBound()**, and the main trail search procedures: **FirstRound()**, **MidRound(·)**, and **FinalRound()**.

EstimateBound() initiates the search by calling **FirstRound()** with an initial weight bound B_{set}. The three main search procedures of $\mathcal{M}$ explore all non-trivial trails whose weights are less than or equal to B_{set}.

If $B_{\text{set}} < \mathbb{B}[R]$, the search cannot yield any valid trail. In that case, **EstimateBound()** incrementally increases the weight bound until $B_{\text{set}} \geq \mathbb{B}[R]$.

In **FirstRound()** and **FinalRound()**, $\mathcal{M}$ proceeds by selecting only the steps that minimize the overall trail weight.

In the **MidRound()** procedure, $\mathcal{M}$ considers all possible trails with non-zero probability, branches the search accordingly, and explores each branch one by one.

Algorithm 3 Matsui's Search Algorithm ($\mathcal{M}$)

Input: $R \geq 2$, $\mathbb{B}[1, \ldots, R-1]$
Output: $\mathbb{T}$ and $\mathbb{B}[R] = W(\mathbb{T})$

$B_{\text{step}} \leftarrow$ constant positive weight
$B_{\text{set}}, \mathbb{T}, \mathbb{T}_{\text{out}}, \text{found} \leftarrow$ False

procedure EstimateBound():
  $B_{\text{set}} \leftarrow \mathbb{B}[R-1]$
  while not found **do**
    $B_{\text{set}} \leftarrow B_{\text{set}} + B_{\text{step}}$
    FirstRound()
  end while
  return $\mathbb{B}[R], \mathbb{T}_{\text{out}}$
end procedure

procedure FirstRound():
  for all $\mathbb{Y} \in \mathbb{F}_2^{wmn} \setminus \{0\}$ **do**
    $\mathbb{T}_1(\mathbb{X}) \leftarrow \arg\min_{\mathbb{X}} W(\mathbb{X} \rightarrow \mathbb{Y})$
    $\mathbb{T}_1(\mathbb{Y}) \leftarrow \mathbb{Y}$
    $\mathbb{T}_1(\mathbb{W}) \leftarrow W(\mathbb{X} \rightarrow \mathbb{Y})$
    if $\mathbb{T}_1(\mathbb{W}) + \mathbb{B}[R-1] \leq B_{\text{set}}$ **then**
      if $R = 2$ **then**
        FinalRound()
      else
        MidRound(2)
      end if
    end if
  end for
end procedure

procedure MidRound(r):
  $\mathbb{T}_r(\mathbb{X}) \leftarrow \text{Lin}(\mathbb{T}_{r-1}(\mathbb{Y}))$
  for all $\mathbb{Y}$ such that $\Pr(\mathbb{T}_r(\mathbb{X}) \rightarrow \mathbb{Y}) \neq 0$ **do**
    $\mathbb{T}_r(\mathbb{Y}) \leftarrow \mathbb{Y}$
    $\mathbb{T}_r(\mathbb{W}) \leftarrow W(\mathbb{T}_r(\mathbb{X}) \rightarrow \mathbb{Y})$
    if $\sum_{i=1}^{r} \mathbb{T}_i(\mathbb{W}) + \mathbb{B}[R-r] \leq B_{\text{set}}$ **then**
      if $r + 1 = R$ **then**
        FinalRound()
      else
        MidRound($r+1$)
      end if
    end if
  end for
end procedure

procedure FinalRound():
  $\mathbb{T}_R(\mathbb{X}) \leftarrow \text{Lin}(\mathbb{T}_{R-1}(\mathbb{Y}))$
  $\mathbb{T}_R(\mathbb{Y}) \leftarrow \arg\min_{\mathbb{Y}} W(\mathbb{T}_R(\mathbb{X}) \rightarrow \mathbb{Y})$
  $\mathbb{T}_R(\mathbb{W}) \leftarrow W(\mathbb{T}_R(\mathbb{X}) \rightarrow \mathbb{Y})$
  if $\sum_{i=1}^{R} \mathbb{T}_i(\mathbb{W}) \leq B_{\text{set}}$ **then**
    found $\leftarrow$ True
    $B_{\text{set}} \leftarrow \sum_{i=1}^{R} \mathbb{T}_i(\mathbb{W})$
    $\mathbb{B}[R] \leftarrow B_{\text{set}}$
    $\mathbb{T}_{\text{out}} \leftarrow \mathbb{T}$
  end if
end procedure

Optimizing Matsui's Search Algorithm

If the left side of the pruning condition becomes bigger, unnecessary trees can be pruned earlier, resulting in a faster search.

4.3 Updated Pruning Condition by BBF15 [2]

Proposition 2. Let $R \geq 1$ and $\mathbb{T}$ be an R-round non-trivial trail. For any $1 \leq r \leq R$ and $0 \leq c < mn$, if $W(\mathbb{T}) \leq \overline{B}$, then

$$\sum_{i=1}^{r-1} \mathbb{T}_i(\mathbb{W}) + \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W}[j]) + \sum_{j=c+1}^{mn-1} \mathbb{T}_r(\mathbb{A}[j]) \times \underline{\mathbb{W}} + B[R-r] \leq \overline{B}$$

where:

- $\underline{\mathbb{W}} = \min\{W(\Delta \xrightarrow{S} \Delta') : \Delta \neq 0\}$ is the minimum possible weight of a non-zero input/output pair through the S-box,
- activity pattern of $\mathbb{T}_i(\mathbb{X})[j]$

$$\mathbb{T}_i(\mathbb{A})[j] = \begin{cases} 1 & \text{if } \mathbb{T}_i(\mathbb{X})[j] \neq 0, \\ 0 & \text{otherwise.} \end{cases}$$

Proof

We begin by expanding the total trail weight:

$$W(\mathbb{T}) = \sum_{i=0}^{R} T_i(\mathbb{W}) = \sum_{i=0}^{r-1} T_i(\mathbb{W}) + T_r(\mathbb{W}) + \sum_{i=r+1}^{R} T_i(\mathbb{W})$$

Since the remaining weight from round $r+1$ to R is at least $\mathbb{B}[R-r]$, we get:

$$\sum_{i=r+1}^{R} T_i(\mathbb{W}) \geq \mathbb{B}[R-r]$$

Thus,

$$W(\mathbb{T}) \geq \sum_{i=0}^{r-1} T_i(\mathbb{W}) + T_r(\mathbb{W}) + \mathbb{B}[R-r]$$

Now, we expand $T_r(\mathbb{W})$ as:

$$T_r(\mathbb{W}) = \sum_{j=0}^{mn-1} \mathbb{T}_r(\mathbb{W})[j] = \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W})[j] + \sum_{j=c+1}^{mn-1} \mathbb{T}_r(\mathbb{W})[j]$$

We define the activity pattern:

$$\mathbb{T}_r(\mathbb{A})[j] = \begin{cases} 1 & \text{if } \mathbb{T}_r(\mathbb{X})[j] \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

Now, consider two cases for each j:

Case 1: $\mathbb{T}_r(\mathbb{X})[j] \neq 0$, then $\mathbb{T}_r(\mathbb{W})[j] \neq 0$ and $\mathbb{T}_r(\mathbb{A})[j] = 1$. Since $\underline{\mathbb{W}}$ is the minimum non-zero S-box weight, we have:

$$\underline{\mathbb{W}} \leq \mathbb{T}_r(\mathbb{W})[j] \quad \Rightarrow \quad \mathbb{T}_r(\mathbb{A})[j] \times \underline{\mathbb{W}} \leq \mathbb{T}_r(\mathbb{W})[j]$$

Case 2: $\mathbb{T}_r(\mathbb{X})[j] = 0$, then $\mathbb{T}_r(\mathbb{W})[j] = 0$ and $\mathbb{T}_r(\mathbb{A})[j] = 0$, so:

$$0 \leq 0 \quad \Rightarrow \quad \mathbb{T}_r(\mathbb{A})[j] \times \underline{\mathbb{W}} \leq \mathbb{T}_r(\mathbb{W})[j]$$

Thus, in both cases:

$$\mathbb{T}_r(\mathbb{A})[j] \times \underline{\mathbb{W}} \leq \mathbb{T}_r(\mathbb{W})[j]$$

Summing over j, we get:

$$T_r(\mathbb{W}) = \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W})[j] + \sum_{j=c+1}^{mn-1} \mathbb{T}_r(\mathbb{W})[j] \geq \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W})[j] + \sum_{j=c+1}^{mn-1} \mathbb{T}_r(\mathbb{A})[j] \times \underline{\mathbb{W}}$$

Plugging this bound back into the earlier inequality:

$$W(\mathbb{T}) \geq \sum_{i=0}^{r-1} T_i(\mathbb{W}) + \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W})[j] + \sum_{j=c+1}^{mn-1} \mathbb{T}_r(\mathbb{A})[j] \times \underline{\mathbb{W}} + \mathbb{B}[R-r]$$

Hence, if $W(\mathbb{T}) \leq \overline{B}$, then the pruning condition becomes:

$$\overline{B} \geq \sum_{i=0}^{r-1} T_i(\mathbb{W}) + \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W})[j] + \sum_{j=c+1}^{mn-1} \mathbb{T}_r(\mathbb{A})[j] \times \underline{\mathbb{W}} + \mathbb{B}[R-r]$$

The pruning condition is checked after the addition of each word to the state in every round. If the condition fails, the algorithm discards the current trail and proceeds to the next possible trail.

4.4 Strengthened Pruning Condition by KHSH22 [3]

4.4.1 Utilizing unbalanced distribution of the minimum weights of S-box

Observe,

$$\mathbb{T}_i(\mathbb{A})[j] \times \underbrace{\underline{\mathbb{W}}}_{\substack{\text{absolute minimum weight} \\ \text{for S-box}}} \leq \underbrace{\min_{Y} W(\mathbb{T}_i(\mathbb{X})[j] \xrightarrow{S} Y)}_{\substack{\text{actual minimum weight possible} \\ \text{for input difference} \\ \mathbb{T}_i(\mathbb{X})[j]}} \leq \mathbb{T}_i(\mathbb{W})[j]$$

The middle term in the above inequality can be a better lower bound for $\mathbb{T}_i(\mathbb{W})[j]$.

Definition 7. Let S be an S-box. If $\min_{\nabla' \neq 0} \mathrm{W}(\nabla \xrightarrow{S} \nabla')$ for each fixed input difference $\nabla \neq 0$ are not same, S has unbalanced differential weights.

The following table presents an example with S-box Sb_0 of MIDORI.

∇	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$\min_{\nabla' \neq 0} \mathrm{W}(\nabla \xrightarrow{S} \nabla')$	2	2	2	2	2	2	2	3	2	2	2	2	2	2	2

For Sb_0 of MIDORI, input difference 8 has slightly higher minimum weight, so Sb_0 has unbalanced difference weight.

If S has unbalanced weights, only few input difference have the minimum differential weight $\underline{\mathbb{W}}$. Then we get strict inequality for many input differences.

$$\mathbb{T}_i(\mathbb{A})[j] \times \underline{\mathbb{W}} < \min_Y W(\mathbb{T}_i(\mathbb{X})[j] \xrightarrow{S} Y)$$

This can be used to strengthen the pruning condition for 2^{nd} and further rounds, as we need the knowledge of full state $\mathbb{T}_i(\mathbb{X})$ to apply this.

Proposition 3. Let $R \geq 1$ and $\mathbb{T}$ be an R-round non-trivial trail. For any $1 \leq r \leq R$ and $0 \leq c < mn$, if $W(\mathbb{T}) \leq \overline{B}$, then:

For $r = 1$:

$$\sum_{j=0}^{c} \mathbb{T}_1(\mathbb{W})[j] + \sum_{j=c+1}^{mn-1} \mathbb{T}_1(\mathbb{A})[j] \times \underline{\mathbb{W}} + B[R-1] \leq \overline{B}$$

For $2 \leq r < R$:

$$\sum_{i=1}^{r-1} \mathbb{T}_i(\mathbb{W}) + \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W})[j] + \sum_{j=c+1}^{mn-1} \min_Y W(\mathbb{T}_r(\mathbb{X})[j] \xrightarrow{S} Y) + B[R-r] \leq \overline{B}$$

Proof

Proof is trivial from Updated Pruning Condition by BBF15 [2] and the fact

$$\mathbb{T}_i(\mathbb{A})[j] \times \underline{\mathbb{W}} \quad \leq \quad \min_Y W(\mathbb{T}_i(\mathbb{X})[j] \xrightarrow{S} Y) \quad \leq \quad \mathbb{T}_i(\mathbb{W})[j]$$

This condition refines the pruning strategy using more precise weight estimations in intermediate rounds.

4.5 Exploiting Branch Number and Structure of Mixing Layer

Branch Number

The branch number of the matrix $\mathbf{M}$ used in Mix layer of AES-like round function, quantifies the mixing power of $\mathbf{M}$. Branch Number is defined as

$$\mathcal{B} = \min_{x \neq 0} \{\text{ACT}(x) + \text{ACT}(M(x))\}$$

where ACT(x) = No. of active words in column x.
The knowledge of the branch number allows us to establish a lower bound on the number of active words in the next round, as stated in the following lemma.

Lemma 1. Let $R \geq 2$ and $\mathbb{T}$ be an R-round non-trivial trail. For any $1 \leq r < R$ and $0 \leq k < n$, the following holds:

$$\left(\sum_{j=0}^{m-1} \mathbb{T}_r(\mathbb{A})[mk+j]\right) \leq \sum_{j=0}^{m-1} \mathbb{T}_{r+1}(\mathbb{A})[\sigma^{-1}(mk+j)],$$

where the function (x) is defined as:

$$(x) = \begin{cases} \max(1, \mathcal{B} - x) & \text{if } x \neq 0, \\ 0 & \text{if } x = 0, \end{cases}$$

and σ is the permutation used in the Shuf layer.

Proof.

From the definition of branch number:

$$\mathcal{B} \leq \mathrm{ACT}(x) + \mathrm{ACT}(M(x)) \quad \forall x \in \mathbb{F}_2^{wm}$$

Let $x = \mathbb{T}_r(\mathbb{Y})\langle k \rangle$, then:

$$M(x) = M(\mathbb{T}_r(\mathbb{Y})\langle k \rangle) = \mathbb{T}_r(\mathbb{Z})\langle k \rangle$$

Now, observe: - If $\mathbb{T}_i(\mathbb{X})[j] = 0$, then $\mathbb{T}_i(\mathbb{Y})[j] = 0$ since the difference propagation $0 \xrightarrow{S} 0$ has probability 1. - If $\mathbb{T}_i(\mathbb{X})[j] \neq 0$, then $\mathbb{T}_i(\mathbb{Y})[j] \neq 0$, as a non-zero input leads to a non-zero output with non-zero probability.

Hence:

$$\mathbb{T}_i(\mathbb{A})[j] = \begin{cases} 0 & \text{if } \mathbb{T}_i(\mathbb{X})[j] = 0 = \mathbb{T}_i(\mathbb{Y})[j] \\ 1 & \text{if } \mathbb{T}_i(\mathbb{X})[j] \neq 0 \neq \mathbb{T}_i(\mathbb{Y})[j] \end{cases}$$

Therefore:

$$\mathrm{ACT}(x) = \mathrm{ACT}(\mathbb{T}_r(\mathbb{Y})\langle k \rangle) = \mathrm{ACT}(\mathbb{T}_r(\mathbb{X})\langle k \rangle) = \sum_{j=0}^{m-1} \mathbb{T}_r(\mathbb{A})[mk+j]$$

$$\mathrm{ACT}(M(x)) = \mathrm{ACT}(M(\mathbb{T}_r(\mathbb{Y})\langle k \rangle)) = \mathrm{ACT}(\mathbb{T}_r(\mathbb{Z})\langle k \rangle)$$

$$= \sum_{j=0}^{m-1} \mathrm{ACT}(\mathbb{T}_r(\mathbb{Z})[mk+j]) = \sum_{j=0}^{m-1} \mathrm{ACT}(\mathbb{T}_r(\mathbb{X}')[\sigma^{-1}(mk+j)])$$

$$= \sum_{j=0}^{m-1} \mathrm{ACT}(\mathbb{T}_{r+1}(\mathbb{X})[\sigma^{-1}(mk+j)]) = \sum_{j=0}^{m-1} \mathbb{T}_{r+1}(\mathbb{A})[\sigma^{-1}(mk+j)]$$

Combining both:

$$\mathcal{B} \leq \sum_{j=0}^{m-1} \mathbb{T}_r(\mathbb{A})[mk+j] + \sum_{j=0}^{m-1} \mathbb{T}_{r+1}(\mathbb{A})[\sigma^{-1}(mk+j)]$$

$$\Rightarrow \mathcal{B} - \sum_{j=0}^{m-1} \mathbb{T}_r(\mathbb{A})[mk+j] \leq \sum_{j=0}^{m-1} \mathbb{T}_{r+1}(\mathbb{A})[\sigma^{-1}(mk+j)]$$

Also, since M is invertible, any non-zero input column always maps to a non-zero output column. Therefore for $x = \mathbb{T}_r(\mathbb{Y})\langle k \rangle \neq 0$:

$$1 \leq \mathrm{ACT}(M(x)) = \sum_{j=0}^{m-1} \mathbb{T}_{r+1}(\mathbb{A})[\sigma^{-1}(mk+j)]$$

Hence:

$$\max\left\{1, \mathcal{B} - \sum_{j=0}^{m-1} \mathbb{T}_r(\mathbb{A})[mk+j]\right\} \leq \sum_{j=0}^{m-1} \mathbb{T}_{r+1}(\mathbb{A})[\sigma^{-1}(mk+j)]$$

$$\Rightarrow \left(\sum_{j=0}^{m-1} \mathbb{T}_r(\mathbb{A})[mk+j]\right) \leq \sum_{j=0}^{m-1} \mathbb{T}_{r+1}(\mathbb{A})[\sigma^{-1}(mk+j)]$$

In the case where $\mathbb{T}_r(\mathbb{Y})\langle k \rangle = 0$, then:

$$\mathrm{ACT}(\mathbb{T}_r(\mathbb{Y})\langle k \rangle) = 0 \Rightarrow \sum_{j=0}^{m-1} \mathbb{T}_r(\mathbb{A})[mk+j] = 0$$

$$\Rightarrow M(\mathbb{T}_r(\mathbb{Y})\langle k \rangle) = 0 \Rightarrow \mathrm{ACT}(M(\mathbb{T}_r(\mathbb{Y})\langle k \rangle)) = 0$$

$$\Rightarrow \sum_{j=0}^{m-1} \mathbb{T}_{r+1}(\mathbb{A})[\sigma^{-1}(mk+j)] = 0$$

So,

$$0 \leq 0 \Rightarrow (0) = 0 \Rightarrow \left(\sum_{j=0}^{m-1} \mathbb{T}_r(\mathbb{A})[mk+j]\right) \leq \sum_{j=0}^{m-1} \mathbb{T}_{r+1}(\mathbb{A})[\sigma^{-1}(mk+j)]$$

Therefore, in either case

$$\left(\sum_{j=0}^{m-1} \mathbb{T}_r(\mathbb{A})[mk+j]\right) \leq \sum_{j=0}^{m-1} \mathbb{T}_{r+1}(\mathbb{A})[\sigma^{-1}(mk+j)]$$

□

Structure of Mixing Layer

We know **Mix** operates as a parallel matrix multiplication **M** column-wise. This structure allows us to determine a lower bound of partial weight of next round, as stated in the following lemma.

Lemma 2. Let $R \geq 2$ and $\mathbb{T}$ be an R-round non-trivial trail. For any $1 \leq r < R$ and $0 \leq c < mn$, it holds that

$$\sum_{k=0}^{n_c-1} \sum_{j=0}^{m-1} \min_{Y} W\left(\mathbb{M}(\mathbb{T}_r(\mathbb{Y})\langle k\rangle[j]) \xrightarrow{S} Y\right) \leq \sum_{j=0}^{mn-1} \mathbb{T}_{r+1}(\mathbb{W})[\sigma^{-1}(j)],$$

where $n_c = \left\lfloor \frac{c+1}{m} \right\rfloor$ and σ is the permutation used in **Shuf**.

Proof. We know $\mathbb{T}_r(\mathbb{Z})\langle k\rangle = \mathrm{M}(\mathbb{T}_r(\mathbb{Y})\langle k\rangle)$ for $0 \leq k < n_c \leq n$,
$\mathbb{T}_{r+1}(\mathbb{X})[j] = \mathbb{T}_r(\mathbb{Z})[\sigma(j)] \Rightarrow \mathbb{T}_{r+1}(\mathbb{X})[\sigma^{-1}(j)] = \mathbb{T}_r(\mathbb{Z})[j] \quad \forall\, 0 \leq j < mn$

$$\begin{aligned}
\sum_{j=0}^{mn_c-1} \mathbb{T}_{r+1}(\mathbb{W})[\sigma^{-1}(j)] &= \sum_{j=0}^{mn_c-1} W(\mathbb{T}_{r+1}(\mathbb{X})[\sigma^{-1}(j)] \xrightarrow{S} \mathbb{T}_{r+1}(\mathbb{Y})[\sigma^{-1}(j)]) \\
&\geq \sum_{j=0}^{mn_c-1} \min_{Y} W(\mathbb{T}_{r+1}(\mathbb{X})[\sigma^{-1}(j)] \xrightarrow{S} Y) \\
&= \sum_{j=0}^{mn_c-1} \min_{Y} W(\mathbb{T}_r(\mathbb{Z})[j] \xrightarrow{S} Y) \\
&= \sum_{k=0}^{n_c-1} \sum_{j=0}^{m-1} \min_{Y} W(\mathbb{T}_r(\mathbb{Z})\langle k\rangle[j] \xrightarrow{S} Y) \\
&= \sum_{k=0}^{n_c-1} \sum_{j=0}^{m-1} \min_{Y} W(M(\mathbb{T}_r(\mathbb{Y})\langle k\rangle)[j] \xrightarrow{S} Y)
\end{aligned}$$

□

4.6 More Strengthened Pruning Condition by KHSH22 [3]

Using Lemma 1 and Lemma 2, a more strengthened pruning condition is derived for non-bit permutation-based AES-like cipher.

Proposition 4. Let $R \geq 2$, $0 \leq c < mn$, and $\mathbb{T}$ be an R-round non-trivial trail over a non-bit-permutation-based AES-like cipher. If $W(\mathbb{T}) \leq \overline{B}$, then the following holds:

for $r = 1$,

$$\sum_{j=0}^{c} \mathbb{T}_1(\mathbb{W})[j] + \mathbb{T}_1(\mathbb{A})[j] \times \underline{\mathbb{W}} + \sum_{k=0}^{n_c-1} \sum_{j=0}^{m-1} \min_{Y} W\left(\mathbb{M}(\mathbb{T}_1(\mathbb{Y})\langle k\rangle[j]) \xrightarrow{S} Y\right) + \sum_{k=n_c}^{n-1} \varphi\left(\sum_{j=0}^{m-1} \mathbb{T}_1(\mathbb{A})[mk+j]\right) \times \underline{\mathbb{W}} + \mathbb{B}[R-2] \leq \overline{B}$$

and for all $2 \leq r < R$,

$$\sum_{i=1}^{r-1} \mathbb{T}_i(\mathbb{W}) + \sum_{j=0}^{c} \mathbb{T}_r(\mathbb{W})[j] + \sum_{j=c+1}^{mn-1} \min_{Y} W\left(\mathbb{T}_r(\mathbb{X})[j] \xrightarrow{S} Y\right) + \sum_{k=0}^{n_c-1} \sum_{j=0}^{m-1} \min_{Y} W\left(\mathbb{M}(\mathbb{T}_r(\mathbb{Y})\langle k\rangle[j]) \xrightarrow{S} Y\right) + \sum_{k=n_c}^{n-1} \varphi\left(\sum_{j=0}^{m-1} \mathbb{T}_r(\mathbb{A})[mk+j]\right) \times \underline{\mathbb{W}} + \mathbb{B}[R-1-r] \leq \overline{B},$$

where

$$\varphi(x) = \begin{cases} \max(1, \mathcal{B} - x) & \text{if } x \neq 0, \\ 0 & \text{if } x = 0, \end{cases} \quad \text{and} \quad n_c = \left\lfloor \frac{c+1}{m} \right\rfloor.$$

Active S-boxes in the First Round

Definition 8. $\mathcal{A}_{tab}$ is the collection of all possible active S-box indices in the first round.

$$\mathcal{A}_{tab} = \{|\mathbb{A}|\text{-combinations of word indices}\{0, 1, \cdots, mn-1\} \quad \text{for } 1 \leq |\mathbb{A}| < mn\}$$

This $\mathcal{A}_{tab}$ is sorted according to ascending order of $|\mathbb{A}|$, i.e. no of active S-boxes.

Example 3. $\mathcal{A}_{tab}$ for **MIDORI** is

$$\{\underbrace{\{0\}, \{1\}, \cdots, \{15\}}_{16 \quad 1-combinations}, \underbrace{\{0,1\}, \{0,2\}, \cdots \{14,15\}}_{120 \quad 2-combinations}, \cdots, \underbrace{\{0,1,2,\cdots,15\}}_{1 \quad 16-combination}\}$$

$|\mathcal{A}_{tab}| = 65535 = 2^{16} - 1$

Using predetermined activity pattern for first round $\mathbb{T}_1(\mathbb{A})$ allows us to obtain a lower bound for weight of first round. To make the search efficient, the algorithm starts by checking activity patterns with fewer active S-boxes first, since those are most likely lead to a best trail.
In the next section, we discuss a property to reduce size of $\mathcal{A}_{tab}$, which will result a faster search, as the algorithm have to go through less trails now.

5 Permutation Characteristic

We observe an important property involving word-wise permutations and the weight of differential transitions in block ciphers.

Proposition 5. Let A be a word-wise permutation. Then for any pair of input and output differences $(\mathbb{X}, \mathbb{Y})$ through a non-linear substitution layer Sub, the following equality holds:

$$W(\mathbb{X} \xrightarrow{\mathsf{Sub}} \mathbb{Y}) = W(A(\mathbb{X}) \xrightarrow{\mathsf{Sub}} A(\mathbb{Y})).$$

Proof. We know

$$\mathbb{X} = [\mathbb{X}[0],\ \mathbb{X}[1],\ \ldots,\ \mathbb{X}[mn{-}1]]$$

Let the word-wise permutation A be defined using $a \in S_{mn}$ as

$$A(\mathbb{X}) = [\mathbb{X}[a(0)],\ \mathbb{X}[a(1)],\ \ldots,\ \mathbb{X}[a(mn{-}1)]]$$

Let $\mathbb{Y} = \mathsf{Sub}(\mathbb{X})$, i.e., $\mathbb{Y}[j] = S(\mathbb{X}[j])$ for all $0 \leq j < mn$.

Then,

$$A(\mathsf{Sub}(\mathbb{X})) = A(\mathbb{Y}) = [\mathbb{Y}[a(0)],\ \mathbb{Y}[a(1)],\ \ldots,\ \mathbb{Y}[a(mn{-}1)]]$$

and

$$\mathsf{Sub}(A(\mathbb{X})) = \mathsf{Sub}([\mathbb{X}[a(0)],\ \ldots,\ \mathbb{X}[a(mn{-}1)]]) = [\mathbb{Y}[a(0)],\ \mathbb{Y}[a(1)],\ \ldots,\ \mathbb{Y}[a(mn{-}1)]]$$

Therefore,

$$A(\mathsf{Sub}(\mathbb{X})) = \mathsf{Sub}(A(\mathbb{X})) \quad \Rightarrow \quad \mathsf{Sub} \circ A = A \circ \mathsf{Sub} \quad \Rightarrow \quad A^{-1} \circ \mathsf{Sub} \circ A = \mathsf{Sub}$$

Recall,

$$W(\mathbb{X} \xrightarrow{\mathsf{Sub}} \mathbb{Y}) = -\log_2 \Pr(\mathbb{X} \xrightarrow{\mathsf{Sub}} \mathbb{Y}) = -\log_2 \left(\frac{|\{P \in 2^{wmn} : \mathsf{Sub}(P \oplus \mathbb{X}) = \mathsf{Sub}(P) \oplus \mathbb{Y}\}|}{2^{wmn}} \right)$$

Now,

$$\begin{aligned} \mathsf{Sub}(P \oplus \mathbb{X}) = \mathsf{Sub}(P) \oplus \mathbb{Y} &\Rightarrow A(\mathsf{Sub}(P \oplus \mathbb{X})) = A(\mathsf{Sub}(P)) \oplus A(\mathbb{Y}) \\ &\Rightarrow \mathsf{Sub}(A(P \oplus \mathbb{X})) = \mathsf{Sub}(A(P)) \oplus A(\mathbb{Y}) \\ &\Rightarrow \mathsf{Sub}(A(P) \oplus A(\mathbb{X})) = \mathsf{Sub}(A(P)) \oplus A(\mathbb{Y}) \end{aligned}$$

Let $R = A(P)$. Since A is a bijection, we have:

$$\mathsf{Sub}(R \oplus A(\mathbb{X})) = \mathsf{Sub}(R) \oplus A(\mathbb{Y})$$

Hence,

$$|\{P \in 2^{wmn} : \mathsf{Sub}(P \oplus \mathbb{X}) = \mathsf{Sub}(P) \oplus \mathbb{Y}\}| = |\{R \in 2^{wmn} : \mathsf{Sub}(R \oplus A(\mathbb{X})) = \mathsf{Sub}(R) \oplus A(\mathbb{Y})\}|$$

Therefore,

$$\begin{aligned} W(\mathbb{X} \xrightarrow{\mathsf{Sub}} \mathbb{Y}) &= -\log_2 \left(\frac{|\{P : \mathsf{Sub}(P \oplus \mathbb{X}) = \mathsf{Sub}(P) \oplus \mathbb{Y}\}|}{2^{wmn}} \right) \\ &= -\log_2 \left(\frac{|\{R : \mathsf{Sub}(R \oplus A(\mathbb{X})) = \mathsf{Sub}(R) \oplus A(\mathbb{Y})\}|}{2^{wmn}} \right) \\ &= W(A(\mathbb{X}) \xrightarrow{\mathsf{Sub}} A(\mathbb{Y})) \end{aligned}$$

□

This property ensures that the differential weight remains unchanged under the application of the same permutation to both input and output differences.

Proposition 6. Let $A, B \in S_{mn}$ be word-wise permutations such that

$$\mathsf{Lin} \circ A = B \circ \mathsf{Lin},$$

where Lin denotes a linear layer. Then for a full round operation $R = \mathsf{Lin} \circ \mathsf{Sub}$, we have:

$$W(\mathbb{X} \xrightarrow{R} \mathbb{X}') = W(A(\mathbb{X}) \xrightarrow{R} B(\mathbb{X}')).$$

Proof. From the previous proof, we know:

$$\mathsf{Sub} \circ A = A \circ \mathsf{Sub} \quad \text{and} \quad \mathsf{Sub} \circ B = B \circ \mathsf{Sub}$$

Now,

$$\begin{aligned} R \circ A &= \mathsf{Lin} \circ \mathsf{Sub} \circ A = \mathsf{Lin} \circ A \circ \mathsf{Sub} \\ &= B \circ \mathsf{Lin} \circ \mathsf{Sub} = B \circ R \end{aligned}$$

Using $R \circ A = B \circ R$,

$$\begin{aligned} R(P \oplus \mathbb{X}) &= R(P) \oplus \mathbb{X}' \\ \Rightarrow B(R(P \oplus \mathbb{X})) &= B(R(P) \oplus \mathbb{X}') \\ \Rightarrow R(A(P \oplus \mathbb{X})) &= B(R(P)) \oplus B(\mathbb{X}') \\ \Rightarrow R(A(P) \oplus A(\mathbb{X})) &= R(A(P)) \oplus B(\mathbb{X}') \end{aligned}$$

Take $S = A(P)$, since A is a bijection:

$$R(S \oplus A(\mathbb{X})) = R(S) \oplus B(\mathbb{X}')$$

$$|\{P \in 2^{wmn} : R(P \oplus \mathbb{X}) = R(P) \oplus \mathbb{X}'\}| = |\{S \in 2^{wmn} : R(S \oplus A(\mathbb{X})) = R(S) \oplus B(\mathbb{X}')\}|$$

Therefore,

$$\begin{aligned} W(\mathbb{X} \xrightarrow{\mathsf{R}} \mathbb{X}') &= -\log_2 \left(\frac{|\{P : R(P \oplus \mathbb{X}) = R(P) \oplus \mathbb{X}'\}|}{2^{wmn}} \right) \\ &= -\log_2 \left(\frac{|\{S : R(S \oplus A(\mathbb{X})) = R(S) \oplus B(\mathbb{X}')\}|}{2^{wmn}} \right) \\ &= W(A(\mathbb{X}) \xrightarrow{\mathsf{R}} B(\mathbb{X}')) \end{aligned}$$

□

Definition 9. A pair of permutations (A, B) is called a **1-round differential permutation characteristic** if

$$\mathsf{Lin} \circ A = B \circ \mathsf{Lin}.$$

We denote this relationship as

$$A \overset{\mathsf{Lin}}{\Longrightarrow} B.$$

Definition 10. A **R-round differential permutation characteristic** is a sequence of word-wise permutations

$$\{D_1,\ D_2,\ \ldots,\ D_{R+1}\}$$

such that

$$D_i \overset{\mathsf{Lin}}{\Longrightarrow} D_{i+1} \quad \text{for } 1 \le i \le R.$$

i.e.

$$D_1 \overset{\mathsf{Lin}}{\Longrightarrow} D_2 \overset{\mathsf{Lin}}{\Longrightarrow} \cdots \overset{\mathsf{Lin}}{\Longrightarrow} D_R \overset{\mathsf{Lin}}{\Longrightarrow} D_{R+1}$$

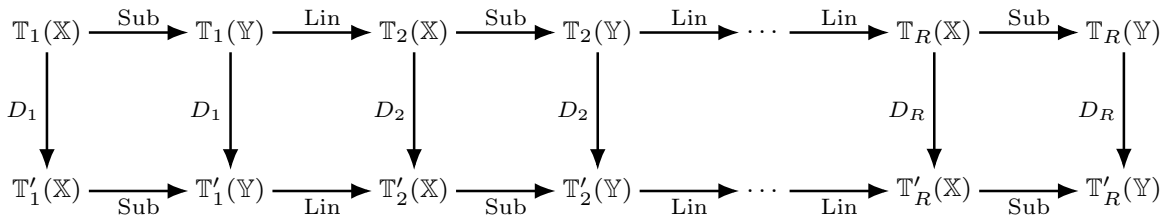
Proposition 7. Let $\mathbb{T}$ be an R-round non-trivial differential trail. If

$$D_1 \overset{\mathsf{Lin}}{\Longrightarrow} D_2 \overset{\mathsf{Lin}}{\Longrightarrow} \cdots \overset{\mathsf{Lin}}{\Longrightarrow} D_{R+1}$$

is an R-round differential permutation characteristic, then the R-round trail $\mathbb{T}'$ defined by

$$\begin{aligned} \mathbb{T}'_i(\mathbb{X}) &= D_i(\mathbb{T}_i(\mathbb{X})), & \mathbb{T}'_i(\mathbb{Y}) &= D_i(\mathbb{T}_i(\mathbb{Y})) \\ \mathbb{T}'_i(\mathbb{Z}) &= \mathsf{Mix}(\mathbb{T}'_i(\mathbb{Y})), & \mathbb{T}'_i(\mathbb{X}') &= D_{i+1}(\mathbb{T}_i(\mathbb{X}')) \end{aligned}$$

for $1 \le i \le R$, is also non-trivial and has the same weight as $W(\mathbb{T})$.



Proof. Clearly, $\mathbb{T}'$ is a non-trivial differential trail.

$$
\begin{aligned}
W(\mathbb{T}') &= \sum_{i=1}^{R} \mathbb{T}'_i(\mathbb{W}) = \sum_{i=1}^{R}\sum_{j=0}^{mn} \mathbb{T}'_i(\mathbb{W})[j] \\
&= \sum_{i=1}^{R}\sum_{j=0}^{mn} W\big(\mathbb{T}'_i(\mathbb{X})[j] \xrightarrow{S} \mathbb{T}'_i(\mathbb{Y})[j]\big) \\
&= \sum_{i=1}^{R}\sum_{j=0}^{mn} W\big(D_i(\mathbb{T}_i(\mathbb{X}))[j] \xrightarrow{S} D_i(\mathbb{T}_i(\mathbb{Y}))[j]\big) \\
&= \sum_{i=1}^{R}\sum_{j=0}^{mn} W\big(\mathbb{T}_i(\mathbb{X})[d_i(j)] \xrightarrow{S} \mathbb{T}_i(\mathbb{Y})[d_i(j)]\big) \\
&= \sum_{i=1}^{R}\sum_{j=0}^{mn} W\big(\mathbb{T}_i(\mathbb{X})[j] \xrightarrow{S} \mathbb{T}_i(\mathbb{Y})[j]\big) \qquad \text{[since } d_i \text{ is a permutation]} \\
&= \sum_{i=1}^{R}\sum_{j=0}^{mn} \mathbb{T}_i(\mathbb{W})[j] = \sum_{i=1}^{R} \mathbb{T}_i(\mathbb{W}) = W(\mathbb{T})
\end{aligned}
$$

□

Reduction of $\mathcal{A}_{\text{tab}}$ Using First-Round Permutations

Let $\mathcal{D}$ be the set of first word-wise permutations D_1 that appear in valid differential permutation characteristics. If two input differences Δ and Δ' are related by a permutation $D \in \mathcal{D}$, i.e., $\Delta' = D(\Delta)$, then any trail starting from Δ can be transformed into an equivalent trail starting from Δ' by applying a permutation characteristic beginning with D to the entire trail.

Therefore, it is sufficient to explore only one representative from each equivalence class induced by the action of $\mathcal{D}$. To avoid redundant computation during trail search, we restrict the activity table $\mathcal{A}_{\text{tab}}$ to contain only these canonical representatives as the first-round input activity patterns. The resulting reduced table is denoted by $\operatorname{Opt}\mathcal{A}_{\text{tab}}$, which significantly prunes the search space without loss of generality, since all equivalent trails can be derived from the stored representatives.

Definition 11. An R-round differential permutation characteristic is called **iterated** if

$$D_1 \overset{\mathsf{Lin}}{\Longrightarrow} D_2 \overset{\mathsf{Lin}}{\Longrightarrow} \cdots \overset{\mathsf{Lin}}{\Longrightarrow} D_{R+1} = D_1.$$

5.1 Computation of Permutation Characteristics

Trivial Permutation Characteristic through Mix

Proposition 8. For any column-wise permutation $C \in S_n$,

$$\text{Mix} \circ C = C \circ \text{Mix} \quad \text{i.e.} \quad C \overset{\text{Mix}}{\Longrightarrow} C.$$

Proof. Let $\mathbb{Y} = [\ \mathbb{Y}\langle 0\rangle, \mathbb{Y}\langle 1\rangle, \ldots, \mathbb{Y}\langle n{-}1\rangle\ ]$ be a state represented as n column vectors. Let $C \in S_n$ be a column-wise permutation, i.e., a permutation that rearranges the positions of these columns.

We apply the linear transformation Mix to the state $\mathbb{Y}$. By definition, this transformation acts independently on each column:

$$\mathsf{Mix}(\mathbb{Y}) = [\ M \cdot \mathbb{Y}\langle 0\rangle,\ M \cdot \mathbb{Y}\langle 1\rangle,\ \ldots,\ M \cdot \mathbb{Y}\langle n{-}1\rangle\ ],$$

where M is the linear mixing matrix.

Applying C to the state $\mathbb{Y}$ means permuting the columns according to C:

$$C(\mathbb{Y}) = [\ \mathbb{Y}\langle C(0)\rangle,\ \mathbb{Y}\langle C(1)\rangle,\ \ldots,\ \mathbb{Y}\langle C(n{-}1)\rangle\ ].$$

Now consider the two compositions:

- Apply C first, then Mix:

$$\mathsf{Mix}(C(\mathbb{Y})) = [M \cdot \mathbb{Y}\langle C(0)\rangle,\ M \cdot \mathbb{Y}\langle C(1)\rangle,\ \ldots,\ M \cdot \mathbb{Y}\langle C(n{-}1)\rangle],$$

- Apply Mix first, then C:

$$C(\mathsf{Mix}(\mathbb{Y})) = [M \cdot \mathbb{Y}\langle C(0)\rangle,\ M \cdot \mathbb{Y}\langle C(1)\rangle,\ \ldots,\ M \cdot \mathbb{Y}\langle C(n{-}1)\rangle].$$

As both compositions result in the same reordered state of mixed columns, we conclude:

$$\mathsf{Mix} \circ C = C \circ \mathsf{Mix}.$$

This equality implies that C commutes with Mix. Therefore, using the notation for differential permutation characteristics:

$$C \overset{\mathsf{Mix}}{\Longrightarrow} C.$$

□

Permutation Characteristic through Mix

$$\begin{aligned}\mathsf{Mix}(\mathbb{X}) &= \mathsf{Mix}\,(\mathbb{X}\langle 0\rangle,\ \mathbb{X}\langle 1\rangle,\ \ldots,\ \mathbb{X}\langle n{-}1\rangle)\\ &= (M\cdot\mathbb{X}\langle 0\rangle,\ M\cdot\mathbb{X}\langle 1\rangle,\ \ldots,\ M\cdot\mathbb{X}\langle n{-}1\rangle)\end{aligned}$$

Therefore, the Mix operation can be represented as a column-wise parallel application of the matrix M:

$$\mathsf{Mix} = (\underbrace{M \parallel M \parallel \cdots \parallel M}_{\text{n times}})$$

Now, suppose for each k such that $0 \le k < n$, we have a pair of permutations A_k and B_k satisfying

$$A_k \overset{M}{\Longrightarrow} B_k \quad \text{i.e.} \quad M \circ A_k = B_k \circ M.$$

Define the parallel compositions:

$$A' = (A_0 \parallel A_1 \parallel \cdots \parallel A_{n-1}), \qquad B' = (B_0 \parallel B_1 \parallel \cdots \parallel B_{n-1}).$$

Then it follows that

$$A' \overset{\mathsf{Mix}}{\Longrightarrow} B'.$$

Proposition 9 (Combining with Previous Results)**.** Let

$$\overline{A} = (A_0 \parallel A_1 \parallel \cdots \parallel A_{n-1}) \circ C, \qquad \overline{B} = (B_0 \parallel B_1 \parallel \cdots \parallel B_{n-1}) \circ C.$$

Then

$$\overline{A} \overset{\mathsf{Mix}}{\Longrightarrow} \overline{B}.$$

5.2 Permutation Characteristic through Lin

For M–S Architecture

Let $\overline{A} \overset{\mathsf{Mix}}{\Longrightarrow} \overline{B}$ and $\mathsf{Lin} = \mathsf{Shuf} \circ \mathsf{Mix}$. Then:

$$\begin{aligned}
& \mathsf{Mix} \circ \overline{A} = \overline{B} \circ \mathsf{Mix}\\
\Longrightarrow\quad & \mathsf{Shuf} \circ \mathsf{Mix} \circ \overline{A} = \mathsf{Shuf} \circ \overline{B} \circ \mathsf{Mix}\\
\Longrightarrow\quad & \mathsf{Shuf} \circ \mathsf{Mix} \circ \overline{A} = \mathsf{Shuf} \circ \overline{B} \circ \mathsf{Shuf}^{-1} \circ \mathsf{Shuf} \circ \mathsf{Mix}\\
\Longrightarrow\quad & \mathsf{Lin} \circ \overline{A} = \mathsf{Shuf} \circ \overline{B} \circ \mathsf{Shuf}^{-1} \circ \mathsf{Lin}\\
\Longrightarrow\quad & \overline{A} \overset{\mathsf{Lin}}{\Longrightarrow} \mathsf{Shuf} \circ \overline{B} \circ \mathsf{Shuf}^{-1}
\end{aligned}$$

For S–M Architecture

Let $\mathsf{Lin} = \mathsf{Mix} \circ \mathsf{Shuf}$ and $\overline{A} \overset{\mathsf{Mix}}{\Longrightarrow} \overline{B}$. Then:

$$
\begin{aligned}
& \mathsf{Mix} \circ \overline{A} = \overline{B} \circ \mathsf{Mix} \\
\Longrightarrow \quad & \mathsf{Mix} \circ \overline{A} \circ \mathsf{Shuf} = \overline{B} \circ \mathsf{Mix} \circ \mathsf{Shuf} \\
\Longrightarrow \quad & \mathsf{Mix} \circ \mathsf{Shuf} \circ \mathsf{Shuf}^{-1} \circ \overline{A} \circ \mathsf{Shuf} = \overline{B} \circ \mathsf{Mix} \circ \mathsf{Shuf} \\
\Longrightarrow \quad & \mathsf{Lin} \circ \mathsf{Shuf}^{-1} \circ \overline{A} \circ \mathsf{Shuf} = \overline{B} \circ \mathsf{Lin} \\
\Longrightarrow \quad & \mathsf{Shuf}^{-1} \circ \overline{A} \circ \mathsf{Shuf} \overset{\mathsf{Lin}}{\Longrightarrow} \overline{B}
\end{aligned}
$$

Permutation Characteristic Search Algorithm

Algorithm 4 1-Round Permutation Characteristics search

Input: Mixing matrix **M**, **Shuf**, cipher's linear layer architecture
Output: $\mathsf{PermChar}(\mathsf{Lin})$

Find all (A, B) such that $M \circ A = B \circ M$, where $A, B \in S_m$, and store them in $\mathsf{PermChar}(M)$.
for each $C \in S_n$ **do**
  for each $\{(A_0, B_0), \ldots, (A_{n-1}, B_{n-1})\} \in \mathsf{PermChar}(M)^n$ **do**
    $\overline{A} \leftarrow (A_0 \parallel \cdots \parallel A_{n-1}) \circ C$
    $\overline{B} \leftarrow (B_0 \parallel \cdots \parallel B_{n-1}) \circ C$
    if cipher architecture is M–S **then**
      $\overline{B} \leftarrow \mathsf{Shuf} \circ \overline{B} \circ \mathsf{Shuf}^{-1}$
      Store $(\overline{A}, \overline{B})$ in $\mathsf{PermChar}(\mathsf{Lin})$
    else if cipher architecture is S–M **then**
      $\overline{A} \leftarrow \mathsf{Shuf}^{-1} \circ \overline{A} \circ \mathsf{Shuf}$
      Store $(\overline{A}, \overline{B})$ in $\mathsf{PermChar}(\mathsf{Lin})$
    end if
  end for
end for

Algorithm 5 R-Round Permutation Characteristics and first-word permutations

Input: No. of rounds $R \geq 2$
Output: $\mathcal{D}$: Set of first-word permutations

$\mathcal{D} \leftarrow \emptyset$
$\overline{\text{PermChar}}(\text{Lin}) \leftarrow \{(A, B) \in \text{PermChar}(\text{Lin}) : \exists X \text{ such that } (B, X) \in \text{PermChar}(\text{Lin})\}$
$G \leftarrow$ a directed graph using pairs in $\overline{\text{PermChar}}(\text{Lin})$ as edges
$\mathcal{C} \leftarrow$ set of connected cyclic subgraphs in G
$\mathcal{L} \leftarrow$ set of connected linear subgraphs in G
for $H \in \mathcal{C}$ **do**
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{\text{all vertices (word-wise permutations) in } H\}$
end for
for $H \in \mathcal{L}$ **do** ▷ Note: one additional vertex is removed from H due to trailing edge
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{\text{first } \max(0, |H| + 1 - R) \text{ vertices in } H\}$
end for
return $\mathcal{D}$

Permutation Matrices and the MIDORI Mixing Layer

Definition 12 (Permutation Matrices)**.** Let $\sigma \in S_4$ be a permutation. The associated permutation matrices are:

- The **row permutation matrix** $R_\sigma = (r_{ij})$ is defined by:

$$r_{ij} = \begin{cases} 1 & \text{if } j = \sigma(i), \\ 0 & \text{otherwise.} \end{cases}$$

- The **column permutation matrix** $C_\sigma = (c_{ij})$ is defined by:

$$c_{ij} = \begin{cases} 1 & \text{if } i = \sigma(j), \\ 0 & \text{otherwise.} \end{cases}$$

Properties of Permutation Matrices. For any matrix $M \in \mathbb{F}^{n\times n}$ and any $\sigma \in S_n$, the following hold:

- $R_\sigma \cdot M$ permutes the rows of M by σ.
- $M \cdot C_\sigma$ permutes the columns of M by σ.
- $R_\sigma^{-1} = R_{\sigma^{-1}}$ and $C_\sigma^{-1} = C_{\sigma^{-1}}$.
- $R_\sigma = C_{\sigma^{-1}}$ and $C_\sigma = R_{\sigma^{-1}}$.

MIDORI Mixing Layer

The mixing matrix of MIDORI is defined as:

$$M_{MIDORI} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} = J - I,$$

where J is the 4×4 all-one matrix and I is the identity matrix.

Commutativity with Permutations. For any 4×4 permutation matrix P, we have:

$$\begin{aligned} M_{\text{MIDORI}} \cdot P &= (J - I) \cdot P = J \cdot P - I \cdot P = J - P, \\ P \cdot M_{\text{MIDORI}} &= P \cdot (J - I) = P \cdot J - P \cdot I = J - P. \\ \therefore \quad M_{\text{MIDORI}} \cdot P &= P \cdot M_{\text{MIDORI}}. \end{aligned}$$

This implies M_{MIDORI} commutes with all permutation matrices of size 4×4.

Implication for Permutation Characteristics. As P is a permutation matrix, there exists $\sigma \in S_4$ such that

$$P = C_\sigma = R_{\sigma^{-1}}$$

$$\therefore \quad M_{MIDORI} \cdot C_\sigma = R_{\sigma^{-1}} \cdot M_{MIDORI} \quad \Rightarrow \quad M_{MIDORI} \circ \sigma = \sigma^{-1} \circ M_{MIDORI}.$$

Thus, permutation characteristics induced by M_{MIDORI} are of the form:

$$\sigma \overset{M_{MIDORI}}{\Longrightarrow} \sigma^{-1}, \quad \text{for } \sigma \in S_4,$$

and we can write:

$$\text{PermChar}(M_{MIDORI}) = \{(\sigma, \sigma^{-1}) : \sigma \in S_4\}.$$

We implemented Algorithm 4 and 5 for few AES-like ciphers and the results are given below

Cipher	| PermChar(M) |	| PermChar(Lin) |	$\lvert\overline{PermChar}(Lin)\rvert$	$\lvert\mathcal{D}\rvert$
AES	4	6144	16	16
MIDORI	24	7962624	576	16
CRAFT	1	24	4	4
LED	1	24	4	4
SKINNY	1	24	4	4

Table 5.1: Permutation Characteristics for Various AES-like Ciphers

6 Conclusion

In this study, we explored Matsui's search algorithm and its optimized versions for AES-like ciphers by strengthening the pruning conditions and employing permutation characteristics. We attempted to create an algorithm to find permutation characteristics for AES-like ciphers. Moreover we applied our algorithm to find permutation characteristics for few AES-like ciphers.

References

[1] M. Matsui, "Linear cryptanalysis method for des cipher," in *Advances in Cryptology — EUROCRYPT '93*, T. Helleseth, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 386–397.

[2] A. Bannier, N. Bodin, and E. Filiol, "Automatic search for a maximum probability differential characteristic in a substitution-permutation network," Cryptology ePrint Archive, Paper 2016/652, 2016. [Online]. Available: https://eprint.iacr.org/2016/652

[3] S. Kim, D. Hong, J. Sung, and S. Hong, "Accelerating the best trail search on AES-like ciphers," Cryptology ePrint Archive, Paper 2022/643, 2022. [Online]. Available: https://eprint.iacr.org/2022/643

[4] E. Biham and A. Shamir, "Differential cryptanalysis of des-like cryptosystems," *Journal of Cryptology*, vol. 4, no. 1, pp. 3–72, Jan 1991. [Online]. Available: https://doi.org/10.1007/BF00630563

[5] M. Matsui, "On correlation between the order of s-boxes and the strength of des," in *Advances in Cryptology — EUROCRYPT'94*, A. De Santis, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 366–375.

[6] S. Banik, A. Bogdanov, T. Isobe, K. Shibutani, H. Hiwatari, T. Akishita, and F. Regazzoni, "Midori: A block cipher for low energy," in *Advances in Cryptology – ASIACRYPT 2015*, T. Iwata and J. H. Cheon, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 411–436.