Automatic Search for a Maximum Probability Differential Characteristic in a Substitution-Permutation Network

Bannier Arnaud, Bodin Nicolas, Filiol Eric
ESIEA Laboratoire (C + V)O, Laval, France
Email: {bannier,bodin,filiol}@esiea-ouest.fr

Abstract—The algorithm presented in this paper computes a maximum probability differential characteristic in a Substitution-Permutation Network (or SPN). Such characteristics can be used to prove that a cipher is practically secure against differential cryptanalysis or on the contrary to build the most effective possible attack. Running in just a few second on 64 or 128-bit SPN, our algorithm is an important tool for both cryptanalysts and designers of SPN.

I. INTRODUCTION

A. Motivation

Modern block ciphers are divided in two categories: Feistel ciphers and Substitution-Permutation Networks, or SPN for short. The encryption process in a Feistel cipher or in a SPN consists in applying a simple operation called round function to the plaintext several times. A different round key is used for each iteration of the round function. These round keys are extracted from a master key using an algorithm called key schedule. Such ciphers are called iterated block ciphers. In a SPN, the round function is made of three distinct stages: a key addition, a substitution layer and a permutation layer.

Differential [1] and linear [2] cryptanalysis are considered as the most important attacks against block ciphers [3]. Therefore, all current ciphers have to resist them. Lai, Massey and Murphy [4] have proposed a formalization for differential cryptanalysis. They clearly exposed the hypothesis made for the attack and introduced Markov ciphers. Since SPN are Markov ciphers, our presentation is based on their work.

A differential predicts that if two plaintexts have a given difference $\alpha$, then the corresponding ciphertexts have a given difference $\beta$ with a certain probability. A differential characteristic is more precise since it gives the difference of intermediate messages for each round. To built a differential cryptanalysis, we usually use a differential that ends just before the last round of the cipher and some ciphertext pairs for which we know that the corresponding plaintexts have the required difference.

For each possible value of the last round key, we decrypt the last round of the ciphertext pairs. If the proportion of obtained pairs satisfying the predicted difference is close to the expected probability, then the chosen round key is probably the right one. Once the last round key is found, it is generally not difficult to recover the entire master key. Therefore, finding an effective differential is the most important part in differential cryptanalysis.

The actual cipher security against differential cryptanalysis is evaluated with the differential probabilities. As these probabilities are difficult to compute, four measures of security have been proposed [5]. They can be split in two categories according to the security they imply.

- The provable security is evaluated by two measures called precise and theoretical. The precise measure gives the maximum differential probability whereas the theoretical measure upper-bounds it.
- The practical security is assessed by two measures called heuristic and practical. The heuristic measure gives the maximum differential characteristic probability while the practical measure upper-bounds it.

The number of chosen plaintexts and the differential cryptanalysis complexity is inversely proportional to these probabilities [1]. A block cipher is said to have provable or practical security whenever these measures are lower than a threshold depending on its features.

It should be emphasized that differential characteristic probabilities are computed assuming that the subkeys are independent and uniformly distributed. Although the subkeys are fixed in a classical differential attack, this assumption provides a good approximation of the true probability. This hypothesis, called stochastic equivalence, seems to hold for almost all secure ciphers. Furthermore, to the authors’ knowledge, there is no practical way to compute the exact probability of a differential.

B. Previous works

Under the previous hypothesis, computing a characteristic probability is simple. However, practical security is assessed by the maximum differential characteristic probability and the number of differential characteristic is such that an exhaustive search is intractable.

In [6], Matsui presented an algorithm that find a maximum probability characteristic in a Feistel cipher. Such an algorithm computes the cipher heuristic measure and enables the practical security evaluation. Running it several times on DES, Matsui found a permutation of the S-boxes making the
DES stronger against both differential and linear cryptanalysis. While its complexity remains high for the cipher FEAL, two successive improvements have been proposed in [7] then [8].

An adaptation of Matsui’s algorithm is possible for SPN. However the block size (64 or 128 bits) of modern ciphers makes the calculations intractable. This fact was also highlighted by Collard et al. [9] who then proposed a few improvements to use this algorithm on the cipher SERPENT. Another variation is exposed by Ali and Heys [10]. They gave up finding a maximum probability characteristic to reduce the complexity. On the other side, their algorithm cannot prove cipher practical security, but may still help the cryptanalyst to build an attack.

C. Contributions

This article presents a search algorithm for a maximal probability differential characteristic in a SPN. Due to the duality between differential and linear cryptanalysis [11], all the results of this article can be adapted to linear cryptanalysis.

The aim of our work is to adapt Matsui’s algorithm for SPN but especially to reduce its complexity greatly. Indeed, spending three months in computing the practical security of a known cipher is not a problem. However, the designer has to repeat several times this search in order to optimize the choice of its cipher components (S-boxes, permutation) or the number of rounds.

In the last few years, many lightweight ciphers have been suggested [12], [13], [14]. They are designed to be implemented in restricted environments such as RFID tags. Consequently, their permutation layers are often bit permutation for efficiency purposes. We have focused our attention on this case and our algorithm allows to analyze practical security of a few cipher systems in just a few seconds.

The security analysis of PRESENT [12], PUFFIN [13] and ICEBERG [14] was performed with the practical measure. Their authors have upper-bounded the probability for a small number of rounds (form 1 to 5) and have then deduced an upper-bound for the full cipher. Our algorithm allows to assess their security more precisely by computing maximum probabilities characteristics.

The following section gives the definitions and notation used in this paper. Section III presents a simple adaptation of Matsui’s algorithm [6]. Our optimizations are exposed in Sections IV and V. Finally, Section VI describes our results.

II. DEFINITIONS

A S-bit substitution box or S-box is a permutation over \( \mathbb{F}_2^S \). A S-Box can be seen as a look-up table. The set of integers from \( a \) to \( b \) included is denoted \([a, b] \).

Definition 1 (SPN). Let \( S \) and \( N \) be positive integers and \( \sigma_1, \ldots, \sigma_N \) be S-bit S-boxes. Let us define the following function

\[
\sigma : (F_2^S)^N \rightarrow (F_2^S)^N
\]

\[
x = (x_1, \ldots, x_N) \rightarrow (\sigma_1(x_1), \ldots, \sigma_N(x_N))
\]

Let \( \pi \) be a bijective \( \mathbb{F}_2 \)-linear mapping from \( \mathbb{F}_2^{SN} \) to \( \mathbb{F}_2^{SN} \). Let us define the round-function \( F \) by

\[
F(k, x) = (\pi \circ \sigma)(x \oplus k)
\]

for any round key \( k \) in \( \mathbb{F}_2^{SN} \) and for any message \( x \) in \( \mathbb{F}_2^{SN} \). The key addition is the operation \( x \mapsto x \oplus k \) which consists of an exclusive OR of the message \( x \) with the round key \( k \).

The functions \( \sigma \) and \( \pi \) are respectively called the substitution layer and the permutation layer of the round function \( F \). An iterated cipher having \( F \) as round-function is called a Substitution-Permutation Network or SPN for short.

Note. The last round of a SPN is usually different from the previous ones. Since a differential characteristic ends just before the last round, this article remains relevant.

Definition 2 (Bit permutation). A linear mapping \( \pi : \mathbb{F}_2^{SN} \rightarrow \mathbb{F}_2^{SN} \) is called bit permutation if there exists a permutation \( \phi \) of \([1, SN]\) such that

\[
\pi(x_1, \ldots, x_{SN}) = (x_{\phi^{-1}(1)}, \ldots, x_{\phi^{-1}(SN)})
\]

Throughout the article, we consider a generic given SPN. The basic aim of differential cryptanalysis [1] is to study the propagation of a difference between two plaintexts \( x_1 \) and \( x_1^* \) through the SPN rounds. Let \((k_1, \ldots, k_R)\) denote fixed round keys used for encryption. For each \( 1 \leq r \leq R \), let us define \( x_{r+1} = F(k_r, x_r) \) and \( x_{r+1}^* = F(k_r, x_r^*) \). The difference \( \alpha_r = x_r \oplus x_r^* \) between \( x_r \) and \( x_r^* \) is fixed by the round key addition since \((x_r \oplus k_r) \oplus (x_r^* \oplus k_r) = x_r \oplus x_r^* = \alpha_r \). Let us denote \( y_r = \sigma(x_r \oplus k_r) \) and \( y_r^* = \sigma(x_r^* \oplus k_r) \) the outputs of the substitution layer and \( \beta_r = y_r \oplus y_r^* \) their difference. Note that \( \beta_r \) is \( k_r \)-dependant. The linearity of \( \pi \) implies that \( \alpha_r+1 = x_{r+1} \oplus x_{r+1}^* = \pi(y_r) \oplus \pi(y_r^*) = \pi(y_r \oplus y_r^*) = \pi(\beta_r) \). Thus, the input difference of the round \( r+1 \) depends only on the output of the round \( r \).

Notation. Input and output differences of the substitution layer for the round \( r \) are respectively referred as

\[
\alpha_r = (a_1^r, \ldots, a_N^r) \quad \text{and} \quad \beta_r = (b_1^r, \ldots, b_N^r)
\]

These belong to \((\mathbb{F}_2^S)^N\). Whenever an arbitrary round is considered, the index \( r \) is omitted and we simply write

\[
\alpha = (a_1, \ldots, a_N) \quad \text{and} \quad \beta = (b_1, \ldots, b_N)
\]

A difference that can be both in input or output of the substitution layer, is denoted \( \gamma = (c_1, \ldots, c_N) \).

In the rest of the paper, the subkeys are assumed to be independent and uniformly distributed. The probability that
a difference $\alpha \in \mathbb{F}_2^S$ produces $\beta \in \mathbb{F}_2^S$ by the $i$-th S-box is given by
\[
P_{i}(\alpha \rightarrow \beta) = \frac{\left| \{ x \in \mathbb{F}_2^S \mid \sigma_{i}(x) \oplus \sigma_{i}(x \oplus a) = b \} \right|}{2^S}.
\]

The $2^S \times 2^S$ matrix formed by these probabilities is called the differential table of the $i$-th S-box. It should be stressed that $P_{i}(0 \rightarrow 0) = 1$. The probability that a difference $\alpha \in (\mathbb{F}_2^S)^N$ produces $\beta$ by the substitution layer is
\[
P(\alpha \rightarrow \beta) = \prod_{i=1}^{N} P_{i}(a_{i} \rightarrow b_{i}),
\]
since the S-boxes are assumed to be independent [4].

**Definition 3** (Active S-box). Let $\gamma$ be a difference. The $i$-th S-box is activated by $\gamma$ if $c_{i} \neq 0$. Let us define the application
\[
\#SB: (\mathbb{F}_2^S)^N \rightarrow \{1, N \}
\]
\[
\gamma \mapsto \left[ \{ i \in \{1, N \} \mid c_{i} \neq 0 \} \right],
\]
that relates a difference to the number of S-Boxes it activates.

**Definition 4** (Candidate). A candidate for an input difference $\alpha$ is an output difference $\beta$ such that $P(\alpha \rightarrow \beta) \neq 0$.

The following lemma links both previous definitions.

**Lemma 5.** If $\beta$ is a candidate for $\alpha$, then $b_{i} = 0 \Rightarrow a_{i} = 0$ for each $i$ such that $1 \leq i \leq N$, that is, they activate the same S-boxes. In this case,
\[
P(\alpha \rightarrow \beta) = \prod_{i, a_{i} \neq 0} P_{i}(a_{i} \rightarrow b_{i}).
\]

**Proof:** Assume that $\beta$ is a candidate for $\alpha$. Let $i$ such that $1 \leq i \leq N$. As $P(\alpha \rightarrow \beta) \neq 0$, we have $P_{i}(a_{i} \rightarrow b_{i}) \neq 0$. As S-boxes are one-to-one, the probability $P_{i}(a_{i} \rightarrow 0)$ is non-zero if and only if $a_{i} = 0$. Further, $P_{i}(0 \rightarrow b_{i})$ is non-zero only if $b_{i} = 0$. The result follows.

**Definition 6** (Characteristic). Let $R$ be a non-negative integer. A $R$-round differential characteristic is an element $\mathcal{T} = ((\alpha_{1}, \beta_{1}), \ldots, (\alpha_{R}, \beta_{R}))$ of $(\mathbb{F}_2^S)^{2R}$ satisfying $\alpha_{r+1} = \pi(\beta_{r})$ for all $1 \leq r < R$. The 0-round characteristic is denoted ( ). For each $0 \leq i \leq j < R$, let $\mathcal{T}_{[i,j]}$ denote the sub-characteristic $((\alpha_{i}, \beta_{i}), \ldots, (\alpha_{j}, \beta_{j}))$.

As we have seen, a difference is fixed by the subkey addition and is mapped by the permutation layer almost surely. The subkeys being independently and uniformly distributed, a $R$-round characteristic probability is computed by
\[
P(\mathcal{T}) = \prod_{r=1}^{R} P_{i}(\alpha_{r} \rightarrow \beta_{r}) = \prod_{r=1}^{R} \left( \prod_{i=1}^{N} P_{i}(a_{i} \rightarrow b_{i}) \right).
\]

**Definition 7** (Optimal characteristic). A $R$-round characteristic with maximum probability among all the $R$-round characteristics is said optimal. In this case, its probability is denoted $p_{\text{Best}(R)}$.

**Definition 8** (Extension). Let $r$ and $r'$ be integers such that $0 \leq r \leq r'$. Let $\mathcal{T}$ and $\mathcal{T}'$ be $r$ and $r'$-round characteristics respectively. The characteristic $\mathcal{T}$ extends $\mathcal{T}'$ if the $r$ first round input and output differences of $\mathcal{T}$ and $\mathcal{T}'$ are equal, that is, $\mathcal{T}_{[1,r]} = \mathcal{T}'$. In this case, $\mathcal{T}' = \mathcal{T} \parallel \mathcal{T}_{[r+1,r']}$. 

**Example 9.** Consider the SPN SMALLPRESENT(4) [15]. Its parameters are $S = 4$ and $N = 4$. The hexadecimal notation is used for the elements of $\mathbb{F}_2^S = \mathbb{F}_4$. For instance, $\text{A} \equiv \{1, 0, 1, 0\}$. The four S-boxes are defined by
\[
\begin{array}{cccccccccccc}
\alpha & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & A & B & C & D & E & F \\
\sigma(i) & C & 5 & 6 & B & 9 & 0 & A & D & 3 & E & F & 8 & 4 & 7 & 1 & 2 \\
\end{array}
\]

The linear layer $\pi$ is the bit permutation associated with $\phi$.

Now we turn our attention to the following pairs of input/output differences
\[
\begin{align*}
\alpha_{1} & = (0, F, 0, 0) & \alpha_{2} & = (0, 0, 0, 4) & \alpha_{3} & = (0, 1, 0, 1) \\
\beta_{1} & = (0, 1, 0, 0) & \beta_{2} & = (0, 0, 0, 5) & \beta_{3} & = (0, 3, 0, 2).
\end{align*}
\]

Since $\alpha_{2} = \pi(\beta_{1})$ and $\alpha_{3} = \pi(\beta_{2})$, they can be concatenated to form longer characteristics. Three rounds of the cipher are represented in the following figure. The S-boxes activated by $\alpha_{1}$, $\alpha_{2}$ and $\alpha_{3}$ are grayed. It is not hard to check that
\[
P_{i}(F \rightarrow 1) = P_{i}(4 \rightarrow 5) = P_{i}(1 \rightarrow 3) = 2^{-2}
\]
and $P_{i}(F \rightarrow 2) = 0$ with (1). Thus, $\beta_{1}$ is a candidate for $\alpha_{1}$ while $(1, 0, 0, 0)$ and $(0, 2, 0, 0)$ are not. Let $\mathcal{T}$ denote the 2-round characteristic $((\alpha_{1}, \beta_{1}), (\alpha_{2}, \beta_{2}))$. Using Equation (2), we have
\[
P(\alpha_{1} \rightarrow \beta_{1}) = \prod_{i=1}^{4} P_{i}(a_{i} \rightarrow b_{i}) = P_{2}(a_{2} \rightarrow b_{2})
\]
\[
= P_{2}(F \rightarrow 1) = 2^{-2}.
\]

Similarly, $P(\alpha_{2} \rightarrow \beta_{2}) = 2^{-2}$ and $P(\alpha_{3} \rightarrow \beta_{3}) = 2^{-4}$. Then $P(\mathcal{T}) = P(\alpha_{1} \rightarrow \beta_{1}) P(\alpha_{2} \rightarrow \beta_{2}) = 2^{-4}$ according to (3). Once the differential table is computed, it is easy to check that $\mathcal{T}$ is optimal as it activates the minimum number of S-boxes with maximum probabilities. There are several optimal characteristics, for instance
\[
\begin{align*}
((0, 0, 0, F), (0, 0, 0, 1)), ((0, 0, 0, 1), (0, 0, 0, 3))
\end{align*}
\]
is also optimal. By definition, $\mathcal{T}' = \mathcal{T} \parallel (\alpha_{3}, \beta_{3})$ extends $\mathcal{T}$. It can be shown that $\mathcal{T}'$ is optimal by running the algorithm presented in this paper. In contrast, $\mathcal{T}'_{[2,3]}$ is not optimal as $P(\mathcal{T}'_{[2,3]}) = 2^{-6} < 2^{-4}$. 

1Let us mention that there may exist more than one optimal characteristic.
III. SEARCH FOR AN OPTIMAL CHARACTERISTIC

A. General principle

Let us denote \( \tilde{R} \) the actual number of rounds. The algorithm presented here computes an optimal \( \tilde{R} \)-round characteristic without requiring any a priori knowledge. It is based on the algorithm OptTrailEst. The latter accepts an integer \( R \geq 2 \), the probabilities \( (p_{\text{Best}}(r))_{1 \leq r < R} \) and an estimation \( p_{\text{Estim}} \) of \( p_{\text{Best}}(R) \) as arguments and returns an optimal \( R \)-round characteristic with its probability \( p_{\text{Best}}(R) \).

The knowledge of \( (p_{\text{Best}}(r))_{1 \leq r < R} \) and \( p_{\text{Estim}} \) speeds up the search. Next, an automatic management of the estimation \( p_{\text{Estim}} \) is proposed which gives rise to the algorithm OptTrail. Thus, OptTrail takes \( R \) and \( (p_{\text{Best}}(r))_{1 \leq r < R} \) as inputs only, and still outputs an optimal \( R \)-round characteristic.

It should be stressed that \( p_{\text{Best}}(1) \) can be easily computed (cf Note in section IV-A). Then, compute

\[
p_{\text{Best}}(R) = \text{OptTrail}\left( R, (p_{\text{Best}}(r))_{1 \leq r < R} \right)
\]

for \( R \) from 2 to \( \tilde{R} \). The latter computation gives the desired result.

In the rest of this section we explain the principle of OptTrailEst. The next section describes several optimizations. Finally, the section V presents the automatic management of the estimation \( p_{\text{Estim}} \).

Figure 1 describes the algorithm OptTrailEst. First, let us suppose that the condition on the bound in procedure Round is true. Under this assumption, the algorithm runs implicitly through the tree of all \( R \)-round characteristics and saves one which has a maximum probability in \( \mathcal{E} \). Observe that the first and last rounds have a special treatment that speeds up the search. When the program reaches the function Round \( (r) \), the current characteristic is \( \mathcal{T} = ((\alpha_1, \beta_1), \ldots, (\alpha_{r-1}, \beta_{r-1})) \) and

\[
\mathbb{P}(\mathcal{T}) = \prod_{i=1}^{r-1} \mathbb{P}(\alpha_i \rightarrow \beta_i) = \prod_{i=1}^{r-1} p_{\text{Estim}}(i).
\]

The input difference \( \alpha_r \) for this round equals \( \pi(\beta_{r-1}) \). Then, for each candidates \( \beta_r \) for \( \alpha_r \), \( \mathcal{T} \) is extended by \( (\alpha_r, \beta_r) \) and the search for the next round is called. Therefore, the program performs a depth-first search. When the program reaches the function LastRound(), it is not hard to compute the output \( \beta_R \) that maximizes the probability of the last round. The characteristic is then saved only if it is better than \( \mathcal{E} \). Let us now explain the condition on the bound.

Definition 10 (Exceeding the bound). Let \( \mathcal{T} \) be a \( r \)-round characteristic with \( r < R \). Its probability \( \mathbb{P}(\mathcal{T}) \) exceeds the rank-\( r \) bound if

\[
\mathbb{P}(\mathcal{T}) > \frac{p_{\text{Estim}}}{p_{\text{Best}}(R-r)}.
\]

This condition on the probability of the current characteristic allows to prune the search tree without missing an optimal characteristic. It can be rewritten \( \mathbb{P}(\mathcal{T}) > p_{\text{Estim}}(R-r) < p_{\text{Estim}} \) and means that even if the characteristic is extended by an optimal \( (R-r) \)-round characteristic, the probability of the whole characteristic would be lower than \( p_{\text{Estim}} \).

The significance of \( p_{\text{Estim}} \) is now clear. If \( p_{\text{Estim}} > p_{\text{Best}}(R) \), a characteristic expandable in an optimal \( R \)-round characteristic can be cut. Furthermore, no characteristic will be saved because of the condition in LastRound(). On the other hand, the closer \( p_{\text{Estim}} \) is from \( p_{\text{Best}}(R) \), the stronger is the pruning condition and the lower is the complexity of OptTrailEst.

B. Proof of the algorithm

We have explained the general principle of the algorithm. Let us now prove the optimality of the characteristic returned.

Lemma 11. Let \( r \) be an integer such that \( 1 \leq r < R \). Let \( \mathcal{T} \) be a \( r \)-round characteristic whose probability exceeds the rank-\( r \) bound. Then there exists no \( R \)-round characteristic extending \( \mathcal{T} \) of probability greater than or equal to \( p_{\text{Estim}} \).

Proof: Assume that \( \mathcal{T}' \) is an extension of \( \mathcal{T} \) such that \( \mathbb{P}(\mathcal{T}') \geq p_{\text{Estim}} \). Then the probability of the \( (R-r) \)-round
Algorithm **OptTrailEst**

For each non-zero output difference $\beta_1$, 
- call the search procedure **FirstRound()**.

If a characteristic has been found ($E$ is not empty), return $E$ and $p_{\text{Estim}}$.

End of the algorithm.

**Function FirstRound()**

$p_{\text{Rd}(1)} \leftarrow \max_{\alpha} \mathbb{P}(\alpha \rightarrow \beta_1)$.

$\alpha_1 \leftarrow \alpha$ such that $\mathbb{P}(\alpha \rightarrow \beta_1) = p_{\text{Rd}(1)}$.

$\alpha_2 \leftarrow \pi(\beta_1)$.

if $R > 2$, then call the search procedure **Round(2)**, else call the search procedure **LastRound()**.

End of the function. (We continue the main loop)

**Function Round(r) (2 \leq r < R)**

For each candidate $\beta_r$ for $\alpha_r$,
- $p_{\text{Rd}(r)} \leftarrow \mathbb{P}(\alpha_r \rightarrow \beta_r)$
- If $\prod_{i=1}^{r} p_{\text{Rd}(i)}$ does not exceed the rank-$r$ bound, then
  - $\alpha_{r+1} \leftarrow \pi(\beta_r)$
  - if $r + 1 < R$, call the search procedure **Round(r + 1)**
  - else, call the search procedure **LastRound()**.

End of the function. (We continue Round(r - 1) or **FirstRound()** if $r = 2$)

**Function LastRound()**

$p_{\text{Rd}(R)} \leftarrow \max_{\beta} \mathbb{P}(\alpha_R \rightarrow \beta)$

$\beta_R \leftarrow \beta$ such that $\mathbb{P}(\alpha_R \rightarrow \beta) = p_{\text{Rd}(R)}$.

If $\prod_{i=1}^{R} p_{\text{Rd}(i)} \geq p_{\text{Estim}}$, then
- $E \leftarrow ((\alpha_1, \beta_1), \ldots, (\alpha_R, \beta_R))$ (this characteristic is saved)
- $p_{\text{Estim}} \leftarrow \prod_{i=1}^{R} p_i = \mathbb{P}(E)$.

End of the function. (We continue Round(R - 1) or **FirstRound()** if $R = 2$)

![Figure 1. Search algorithm for an optimal characteristic](image)

characteristic $\mathcal{T}'_{r+1,R}$ is

$$
\mathbb{P}(\mathcal{T}'_{r+1,R}) = \frac{\mathbb{P}(\mathcal{T} \mathcal{T}'_{r+1,R})}{\mathbb{P}(\mathcal{T})} = \frac{\mathbb{P}(\mathcal{T} \mid \mathcal{T}'_{r+1,R})}{\mathbb{P}(\mathcal{T})} = \frac{\mathbb{P}(\mathcal{T}')}{\mathbb{P}(\mathcal{T})}.
$$

By assumption, $\mathbb{P}(\mathcal{T}) < p_{\text{Estim}} / p_{\text{Best}(R-r)}$ holds. Note that this strict inequality implies $p_{\text{Estim}} > 0$. It follows that

$$
\frac{\mathbb{P}(\mathcal{T}')}{\mathbb{P}(\mathcal{T})} > \frac{p_{\text{Estim}}}{p_{\text{Estim}} / p_{\text{Best}(R-r)}} = p_{\text{Best}(R-r)}.
$$

By definition of $p_{\text{Best}(R-r)}$, this leads to a contradiction and hence the result follows.

**Theorem 12** (Validity of the algorithm). The algorithm **OptTrailEst** returns a characteristic $E$ such that $\mathbb{P}(E) = p_{\text{Best}(R)}$ if there exists a $R$-round characteristic of probability greater than $p_{\text{Estim}}$, in other words, if $p_{\text{Estim}} \leq p_{\text{Best}(R)}$. Otherwise, the algorithm does not return anything.

**Proof:** Suppose the condition on the bound removed. If $p_{\text{Estim}}$ is lower than $p_{\text{Best}(R)}$, an optimal characteristic is saved in $E$, otherwise $E$ remains empty. The previous Lemma ensures that the pruning condition avoids only characteristic with probability strictly lower than $p_{\text{Estim}}$. The result still holds.

**IV. Optimizations**

The complexity of this OptTrailEst version of is too large to be achievable with real-sized SPN. For example, the first step requires to call the procedure FirstRound for all non-zero output differences $\beta_1$. Since there are $2^{SN} - 1$ such differences, we can lower-bound its complexity by $2^{64}$ if $N = 16$, $S = 4$ and by $2^{128}$ if $N = 16$, $S = 8$. The optimization of the different parts is the focus of this section.

**A. Construction of the first difference**

As we have just seen, the number of calls to the function FirstRound() is a problem that we must now solve. To optimize this step, a partition of the set of all non-zero
differences is defined. Then, we give an effective way to test whether no difference in one part can be the beginning of an optimal characteristic.

Let \( n \) be an integer such that \( 1 \leq n \leq N \). The maximum probability of the \( n \)-th S-box is

\[
p_{SB(i)} = \max_{a,b} \Pr((a \rightarrow b)).
\]

Let us sort these probabilities in the decreasing order. This is equivalent to define a permutation \( \tau \) of \([1,N]\) such that \( p_{SB(\tau(i))} \geq p_{SB(\tau(i+1))} \) for all \( 1 \leq i < N \). Let \( p_{[n]\cdot SB} \) denote the maximum probability of a one-round characteristic activating \( n \) S-boxes. In other words we have

\[
p_{[n]\cdot SB} = \max_{\#SB(\alpha) = n} \Pr(\alpha \rightarrow \beta).
\]

**Proposition 13.** Let \( n \) be an integer such that \( 1 \leq n \leq N \). Then,

\[
p_{[n]\cdot SB} = \prod_{i=1}^{n} p_{SB(\tau(i))}
\]

**Proof:** Let \( \alpha \) be an input difference activating \( n \) S-boxes and \( \beta \) be an output difference. We will prove that \( \prod_{i=1}^{n} p_{SB(\tau(i))} \geq \Pr(\alpha \rightarrow \beta) \). For each \( i \) in \([1,N]\), define \( q_i = \Pr((a_i \rightarrow b_i)) \). By definition, \( \Pr(\alpha \rightarrow \beta) = \prod_{i=1}^{N} \Pr(a_i \rightarrow b_i) = \prod_{i=1}^{N} q_i \). Let \( \rho \) be a permutation of \([1,N]\) such that \( q_\rho(i) \geq q_{\rho(i+1)} \). Since the input difference \( \alpha \) activates \( n \) S-boxes, it must be the case that \( q_\rho(i) = 0 \) for each \( i > n \). It follows that \( \Pr(\alpha \rightarrow \beta) = \prod_{i=1}^{N} q_i = \prod_{i=1}^{n} q_{\rho(i)} \).

As \( \prod_{i=1}^{n} p_{SB(\tau(i))} \) is the product of the \( n \) best probabilities, the inequality \( \prod_{i=1}^{n} p_{SB(\tau(i))} \geq \prod_{i=1}^{n} p_{SB(\rho(i))} \) holds. Next, \( \prod_{i=1}^{n} p_{SB(\rho(i))} \geq \prod_{i=1}^{n} q_{\rho(i)} \) since \( p_{SB(i)} \geq q_i \geq 0 \) for each \( i \) in \([1,N]\). The result hence follows.

This proposition makes the computation of \( p_{[n]\cdot SB} \) easy as the probabilities \( p_{SB(i)} \) can be read on the differential tables.

**Note.** As a corollary, the inequalities \( p_{[1]\cdot SB} \geq \ldots \geq p_{[N]\cdot SB} \) hold. Thus, the probability of an optimal one-round characteristic is \( p_{est(1)} = \max_{\alpha,\beta} \Pr(\alpha \rightarrow \beta) = p_{[1]\cdot SB} = p_{SB(\tau(1))} \). Of course, the differential tables and the probabilities \( p_{SB(i)} \) and \( p_{[i]\cdot SB} \) are precomputed to optimize the search.

**Theorem 14.** Let \( n \) and \( n' \) be integers such that \( 1 \leq n \leq n' \leq N \). If \( p_{[n]\cdot SB} \) exceeds the rank-one bound, then there exists no \( R \)-round characteristic activating \( n' \) S-boxes in the first round with probability greater than or equal to \( p_{est} \).

**Proof:** Assume that \( p_{[n]\cdot SB} \) exceeds the rank-one bound. Let \( T \) be a one-round characteristic activating \( n' \) S-boxes. By definition, \( \Pr(T) \leq p_{[n']\cdot SB} \). The inequality \( p_{[n']\cdot SB} \leq p_{[n]\cdot SB} \) follows from Proposition 13, therefore \( \Pr(T) \leq p_{[n]\cdot SB} \). Hence, \( \Pr(T) \) exceeds the rank-one bound and Lemma 11 ensures that there exists no \( R \)-round characteristic extending \( T \) with probability greater than or equal to \( p_{est} \). This concludes the proof.

---

**Algorithm OptTrailEst**

For \( n \) from 1 to \( N \),

- if \( p_{[n]\cdot SB} \) exceeds the rank-one bound, then exit the loop (see Theorem 14);
- else,
  - for each output difference \( \beta_1 \) activating \( n \) S-boxes,
    - call FirstRound().

If a characteristic has been found (\( \mathcal{E} \) is not empty), return \( \mathcal{E} \) and \( p_{est} \).

Figure 2. First optimization – construction of the first difference

This theorem states that whenever \( p_{[n]\cdot SB} \) exceeds the rank-one bound, we only have to test the output differences \( \beta_1 \) activating at most \( n - 1 \) S-boxes. There are

\[
\sum_{i=1}^{n-1} \binom{N}{i} (2^S - 1)^i
\]

such differences, compared with \( 2^{SN} - 1 \) otherwise.

We have run the final algorithm with several SPN having a bit permutation as linear layer. With \( N = 16 \) and \( S = 4 \), \( p_{[4]\cdot SB} \) always exceeded the rank-one bound. There are at most \( 2^{21} \) differences to be tested instead of \( 2^{64} \). With \( N = 16 \) and \( S = 8 \), the gap is even larger since \( p_{[3]\cdot SB} \) always exceeds the bound. Finally, \( 2^{21} \) differences instead of \( 2^{128} \) remain to be tested. The algorithm optimized with Theorem 14 is described in Figure 2.

---

**B. The round function**

Following Matsui’s algorithm [6], the round candidates are constructed recursively. Let \( \alpha \) denote the input difference of the current round. According to Lemma 5, a candidate \( \beta \) is constructed by selecting an output for each S-box activated by \( \alpha \).

Recall that the support of \( L \) in \( F_2^{2N} \) is the set \( \text{supp}(L) = \{i \in [1,N] \mid L_i \neq 0\} \) and its Hamming weight is \( \omega(L) = |\text{supp}(L)| \).

The function \( \text{list} : (F_2^S)^N \rightarrow (F_2^N)^N \) maps a difference \( \gamma \) to the vector \( \text{list(\gamma)} = (x_1, \ldots, x_N) \) where \( x_i \) equals 1 if and only if the \( i \)-th S-box is activated by \( \gamma \). It is clear that \#SB(\gamma) = \omega(\text{list(\gamma)}). Further, \( \beta \) is a candidate for \( \alpha \) if and only if \( \text{list(\alpha)} = \text{list(\beta)} \).

Let \( L \in F_2^{2N} \) be a compact representation of active S-boxes and define

\[
p_{\text{ListSB}(L)} = \prod_{i \in \text{supp}(L)} p_{SB(i)}.
\]

Let \( \lor \) denote the bitwise OR and \( \land \) denote the bitwise AND. Next, the vector of size \( n \) in which the first \( n \) coordinates are 1 and the last \( N-n \) are 0 is denoted \((0_n, 1_{N-n})\).
Example 15. With the same notations as in the previous example, \( \text{list}(\alpha_3) = \text{list}(\beta_3) = (0, 1, 0, 1) \) and \( \text{list}(\alpha_3) \land (0_2) = (0, 0, 1, 1) \). Define the function \( \rho : [1, n] \to [1, N] \) that relates \( i \) to the index of the \( i \)-th S-box activated by \( \alpha_r \). Finally, let us define \( L = \text{list}(\alpha_r) \land (0_1, 0_2, n_1) \). If

\[
\Pr\left( T_{[1,1]r} \right) \prod_{i=1}^{n} \Pr(\rho(i)) a_{\rho(i)}^r \to b_{\rho(i)}^r \right) p_{\text{ListSB}}(L)
\]

exceeds the \( r \)-rank bound, then for all \( \gamma \) satisfying:

- \( c_{\rho(i)} = b_{\rho(i)}^r \) for each \( i < n \), and
- \( \Pr(\rho(n)) a_{\rho(n)}^r \to c_{\rho(n)} \leq \Pr(\rho(n)) a_{\rho(n)}^r \to b_{\rho(n)}^r \),

there exists no R-round characteristic extending \( T_{[1,1]r} \| (\alpha_r, \gamma) \) with probability greater than or equal to \( \Pr(\text{Estim}) \).

Proof: If \( \gamma \) is not a candidate for \( \alpha_r \), then \( \Pr(\alpha_r \to \gamma) = 0 \) and any characteristic extending \( T_{[1,1]r} \| (\alpha_r, \gamma) \) has also a zero probability. Therefore, we assume that \( \gamma \) is a candidate for \( \alpha_r \) in the following. Since \( \text{supp}(L) = \{\rho(i) | n + 1 \leq i \leq m\} \), it follows that:

\[
\Pr(\alpha_r \to \gamma) = \prod_{i=1}^{m} \Pr(\rho(i)) a_{\rho(i)}^r \to c_{\rho(i)}
\]

\[
= \prod_{i=1}^{n} \Pr(\rho(i)) a_{\rho(i)}^r \to c_{\rho(i)} \prod_{i=n+1}^{m} \Pr(\rho(i)) a_{\rho(i)}^r \to c_{\rho(i)}
\]

\[
\leq \prod_{i=1}^{n} \Pr(\rho(i)) a_{\rho(i)}^r \to b_{\rho(i)}^r \prod_{i=n+1}^{m} \Pr(\rho(i)) a_{\rho(i)}^r \to c_{\rho(i)}
\]

\[
\leq \prod_{i=1}^{n} \Pr(\rho(i)) a_{\rho(i)}^r \to b_{\rho(i)}^r \right) p_{\text{ListSB}}(L)
\]

Next, we have the inequality:

\[
\Pr\left( T_{[1,1]r} \| (\alpha_r, \gamma) \right) \geq \prod_{i=1}^{n} \Pr(\rho(i)) a_{\rho(i)}^r \to b_{\rho(i)}^r \right) p_{\text{ListSB}}(L)
\]

Consequently, the probability of \( T_{[1,1]r} \| (\alpha_r, \gamma) \) exceeds the \( r \)-rank bound. The result then is a consequence of Lemma 11. \( \square \)

Note. All the probabilities \( p_{\text{ListSB}}(L) \), \( L \in \mathbb{P}_N^2 \) are precomputed. For each \( i \in [1, N] \) and each input difference \( a \), the output differences are sorted in decreasing order.

C. Active S-boxes in the next round

Throughout this part, the linear layer \( \pi \) is assumed to be a bit permutation.

Definition 17. Let \( L \) and \( L' \) be elements of \( \mathbb{P}_N^2 \). Define \( L \leq L' \) if and only if \( \text{supp}(L) \subseteq \text{supp}(L') \). It is easy to show that \( \leq \) is partial order. Clearly:

\[
(L \leq L') \implies (p_{\text{ListSB}}(L') \leq p_{\text{ListSB}}(L)).
\]

Denote \( 0_S = (0, \ldots, 0) \) the identity element of \( \mathbb{P}_N^2 \). Let us define the function \( D : [1, N] \times \mathbb{P}_N^2 \to (\mathbb{F}_2^N)^N \) that relates a pair \((i, c)\) to \( D(i, c) = (c_1, \ldots, c_N) \) where \( c_j \) equals \( c \) if \( i = j \) and \( 0 \) otherwise. To simplify, let us denote \( D_i(c) = D(i, c) \).

Example 18. Using again the previous notation, \( \beta_2 = (0, 0, 0, 0) \) holds. \( \beta_3 = \sum_{i} a_i \beta_4 D_i(b_i^1) = D_2(3) + D_4(3) \).

Lemma 19. Let \( \gamma_1, \ldots, \gamma_n \) be n differences mutually disjoint. Then:

\[
\Pr\left( \bigvee_{i=1}^{n} \gamma_i \right) = \Pr\left( \bigvee_{i=1}^{n} \text{list}(\gamma_i) \right).
\]

Proof: By induction on \( n \). The case \( n = 1 \) is trivial, we assume \( n = 2 \). Define \( L = \text{list}(\gamma_1 + \gamma_2) \), \( L' = \text{list}(\gamma_1) \) and \( L'' = \text{list}(\gamma_2) \). Let \( i \) be an integer such that \( 1 \leq i \leq n \).

Since \( \gamma_1 \) and \( \gamma_2 \) are mutually disjoint, the equality \( c^1_i + c^2_i = 0 \) implies \( c^1_i = 0 \) and \( c^2_i = 0 \). The converse being immediate, the equivalence \( (c^1_i + c^2_i = 0) \iff (c^1_i = 0 \land c^2_i = 0) \) follows, that is \( (c^1_i + c^2_i = 0) \iff (c^1_i \neq 0 \lor c^2_i \neq 0) \).

Next, \( L_1 = 1 \Rightarrow c^1_i + c^2_i = 0 \iff (c^1_i \neq 0 \lor c^2_i = 0) \iff (L_1 = 1 \land L_2 = 1) \).

Therefore, \( L = L' \lor L'' \). The result follows by induction on \( n \) as \( \gamma_n \) and \( \sum_{i=1}^{n} \gamma_i \) are mutually disjoint. \( \square \)

Corollary 20. Let \( \beta \) be an output difference. Let \( m \) be an integer such that \( 1 \leq m \leq N \) and \( \rho : [1, m] \to [1, N] \) an one-to-one function. Then:

\[
\Pr\left( \bigvee_{i=1}^{m} \text{list}(\pi(D_\rho(i)(b_\rho(i))) \right) = \Pr\left( \bigvee_{i=1}^{m} \text{list}(\pi(D_\rho(i)(b_\rho(i))) \right).
\]

Proof: Since \( \pi \) is linear, it follows that the equality \( \pi(S_{i=1}^{m} D_\rho(i)(b_\rho(i))) = \sum_{i=1}^{m} \pi(D_\rho(i)(b_\rho(i))) \) holds. Clearly, the \( D_\rho(i)(b_\rho(i)) \) are mutually disjoint as \( \rho \) is one-to-one. Since \( \pi \) is a bit permutation, it must be the case that the \( \pi(D_\rho(i)(b_\rho(i))) \) are also disjoint. From Lemma 19, we have:

\[
\Pr\left( \bigvee_{i=1}^{m} \pi(D_\rho(i)(b_\rho(i))) \right) = \Pr\left( \bigvee_{i=1}^{m} \text{list}(\pi(D_\rho(i)(b_\rho(i))) \right).
\]

Example 21. On the one hand, \( \text{list}(\pi(D_1(c) + D_2(9))) = \text{list}(\pi(C, 9, 0, 0)) = \text{list}(C, 8, 0, 4) = (1, 1, 0, 1) \).

On the other hand:

\[
\text{list}(\pi(D_1(c))) \lor \text{list}(\pi(D_2(9)))
\]

\[
= \text{list}(8, 8, 0, 0) \lor \text{list}(4, 0, 0, 4)
\]

\[
= (1, 1, 0, 0) \lor (1, 0, 0, 1) = (1, 1, 0, 1) \).
We use the same notation as in Theorem 16 with probability \( i \leq R \| \pi \leq \gamma \) is a candidate for \( \gamma_n + 1 \) for each list \( \alpha = D \) such that \( \beta \) be an output difference. 

\[ \text{else, call } \text{SubRound}(r, n) \text{ or LastRound()} \text{ if } r = 2 \]

\[ \text{End of the function. (We continue Round(r - 1) or FirstRound() if } r = 2) \]

**Function SubRound \((r, n)\)**

if \( n > \#SB(\alpha_r) \),
- \( p_{\text{Rad}}(r) \leftarrow \mathbb{P}(\alpha_r \rightarrow \beta_r) \cdot \prod_{j=1}^{\#SB(\alpha_r)} p_{\text{Rad}}(p_r) \); 
- \( \alpha_{r+1} \leftarrow \pi(\beta_{r+1}) \); 
- if \( r + 1 < R \), then call \( \text{Round}(r + 1) \) else call \( \text{LastRound()} \); 
- \( \text{else, call } \text{SubRound}(r, n + 1) \).

\[ \text{End of the function. (We continue SubRound}(r, n - 1) \text{ or Round}(r)) \]

**Theorem 22.** We use the same notation as in Theorem 16 except that \( r \leq R - 1 \). Define \( L' = \bigvee_{i=1}^{n} \text{list}(\pi(D_{\rho}(b_{\rho(i)}^r))) \).

\[ \mathbb{P}(\text{T}_{[1, r]}(\prod_{i=1}^{n} \mathbb{P}_{\rho}(a_{\rho(i)}^r \rightarrow b_{\rho(i)}^r)) \cdot p_{\text{ListSB}(L)}) \]

\[ \times p_{\text{ListSB}(L')} \]

exceeds the rank-(\( r + 1 \)) bound, then for all \( \gamma \) such that \( c_{\rho(i)} = b_{\rho(i)}^r \) for each \( i \leq n \), there exists no \( R \)-round characteristic extending \( \text{T}_{[1, r]}(\prod_{i=1}^{n} \mathbb{P}_{\rho}(a_{\rho(i)}^r \rightarrow b_{\rho(i)}^r)) \) with probability greater than or equal to \( p_{\text{ListSB}(L')} \).

**Proof:** Following the proof of Theorem 16, we can assume that \( \gamma \) is a candidate for \( \alpha_r \) and deduce the upper-bound

\[ \mathbb{P}(\text{T}_{[1, r]}(\prod_{i=1}^{n} \mathbb{P}_{\rho}(a_{\rho(i)}^r \rightarrow b_{\rho(i)}^r)) \cdot p_{\text{ListSB}(L)} \cdot \mathbb{P}(\omega(\gamma)) \]

\[ \leq \mathbb{P}(\text{T}_{[1, r]}(\prod_{i=1}^{n} \mathbb{P}_{\rho}(a_{\rho(i)}^r \rightarrow b_{\rho(i)}^r)) \cdot p_{\text{ListSB}(L')} \cdot \mathbb{P}(\omega(\gamma)) \]

Define \( \alpha_{r+1} = \pi(\gamma) \). Let \( \beta_{r+1} \) be an output difference. Similarly, we can assume that \( \beta_{r+1} \) is a candidate for \( \alpha_{r+1} \). Define

\[ L'' = \text{list}(\pi(\sum_{i=1}^{n} D_{\rho}(c_{\rho(i)}))) = \text{list}(\pi(\gamma)) = \text{list}(\alpha_{r+1}) \cdot \]

Since \( L'' = \text{list}(\alpha_{r+1}) \), it follows that \( \mathbb{P}(\alpha_{r+1} \rightarrow \beta_{r+1}) \leq \mathbb{P}_{\text{ListSB}(L')} \). By Corollary 20,

\[ \mathbb{P}(\text{T}_{[1, r]}(\prod_{i=1}^{n} \mathbb{P}_{\rho}(a_{\rho(i)}^r \rightarrow b_{\rho(i)}^r)) \cdot p_{\text{ListSB}(L')}) \]

\[ = \mathbb{P}(\text{T}_{[1, r]}(\prod_{i=1}^{n} \mathbb{P}_{\rho}(a_{\rho(i)}^r \rightarrow b_{\rho(i)}^r)) \cdot p_{\text{ListSB}(L)}) \]

Thus, the probability of \( \text{T}_{[1, r]}(\prod_{i=1}^{n} \mathbb{P}_{\rho}(a_{\rho(i)}^r \rightarrow b_{\rho(i)}^r)) \) exceeds the rank-(\( r + 1 \)) bound and there exists no \( R \)-round characteristic extending it with probability greater than or equal to \( p_{\text{ListSB}(L')} \). Using the fact that this property holds for all \( \beta_{r+1} \), the desired result is proven.

The search procedure Round optimized with Theorems 16 and 22 is described in Figure 3.

**D. Test on the bound**

The previous results can be preserved while strengthening the condition on the bound. Suppose we have a characteristic
with probability greater than or equal to \( p_{\text{Estim}} \) found. Then, we have \( p_{\text{Estim}} = \mathbb{P}(E) \). Now, assume that the probability of the current characteristic \( T \) satisfies \( \mathbb{P}(T) \cdot p_{\text{Best}(r-T)} = p_{\text{Estim}} \). In this case, this probability does not exceed the rank-\( r \) bound and the algorithm tries to extend it. However, the previous equality implies that we can optimally find a \( R \)-round characteristic with probability \( p_{\text{Estim}} \). As such a characteristic is already known \( (E) \), the extension of \( T \) can be aborted. This discussion leads us to improve Definition 10.

**Definition 23** (Exceeding the bound). Let \( T \) be a \( r \)-round characteristic with \( r < R \). Its probability \( \mathbb{P}(T) \) exceeds the rank-\( r \) bound if

- \( E \) is empty and
- \( \mathbb{P}(T) < p_{\text{Estim}} / p_{\text{Best}(r-T)} \).

or if

- \( E \) contains a characteristic and
- \( \mathbb{P}(T) \leq p_{\text{Estim}} / p_{\text{Best}(r-T)} \).

V. AUTOMATIC MANAGEMENT OF THE ESTIMATION

The parameter \( p_{\text{Estim}} \) has an important impact on the complexity of OptTrailEst. Several methods exist to obtain a good estimation of \( p_{\text{Best}(R)} \). For instance, an iterative characteristic can be used. Following an idea of Ohta, Moriai and Aoki [7], we propose the algorithm OptTrail. The latter has two main advantages. First, the estimation management is fully automatic – no knowledge is required on the SPN. Second, its complexity has the same order of magnitude, as OptTrailEst runs with \( p_{\text{Estim}} = p_{\text{Best}(R)} / 2 \).

The algorithm OptTrail is presented in Figure 4. To understand how it works, it is worth recalling that OptTrailEst finds no characteristic whenever \( p_{\text{Estim}} > p_{\text{Best}(R)} \) (Theorem 12). In this case, \( p_{\text{Estim}} \) is not modified by OptTrailEst. Since \( p_{\text{Best}(R)} < p_{\text{Best}(R-1)} \) (the probability of a one-round characteristic is strictly lower than 1 for an SPN provided the S-boxes are non-linear), we begin by running OptTrailEst with \( p_{\text{Estim}} = p_{\text{Best}(R-1)} / 2 \). The estimation is then each time divided by two until an optimal characteristic is found. This happens whenever the condition \( p_{\text{Estim}} \leq p_{\text{Best}(R)} \) becomes true.

Indeed, the larger is the value of \( p_{\text{Estim}} \), the stronger is the pruning condition and the lower is the complexity of the search. The exact nature of this result is still unknown. However, we have observed experimentally that the complexity of OptTrailEst running with \( p_{\text{Estim}} \geq 2^4 \cdot p_{\text{Best}(R)} \) is negligible compared with the complexity of the same algorithm running with \( p_{\text{Estim}} = p_{\text{Best}(R)} / 2 \). The following result comes from this observation: if OptTrailEst is computable with \( p_{\text{Estim}} = p_{\text{Best}(R)} / 2 \), then OptTrail is also computable.

**Proposition 24.** The complexity of OptTrailEst decreases as the input \( p_{\text{Estim}} \) increases.

VI. RESULTS

Experiments and simulations have been performed by a AMD Phenom II X4 965 Black Edition 3.4 GHz processor. The running time for a \( R \)-round cipher includes the pre-computations and \( R - 1 \) calls to OptTrail, as explained in Section III.

To prove the practical security of PRESENT [12] against differential cryptanalysis, the authors have shown that the probability of any 5-round characteristic is upper-bounded by \( 2^{-20} \) and had exhibited a 5-round characteristic of probability \( 2^{-21} \). The algorithm presented here allows us to prove in 0.3 second that this upper-bound is reached with the following optimal characteristic:

\[
\begin{align*}
\alpha_1 &= (0,0,0,7,0,0,0,0,0,0,0,0,0,0,0,0,0) \\
\beta_1 &= (0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,1) \\
\beta_2 &= (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3) \\
\beta_3 &= (0,0,0,0,0,0,0,0,0,0,0,0,0,4,0,0,0,4) \\
\beta_4 &= (0,0,0,0,0,0,3,0,0,0,0,0,0,0,0,0,0) \\
\beta_5 &= (0,0,0,0,0,0,0,0,0,6,0,0,0,6,0,0,0) \\
\end{align*}
\]

They have then deduced that any 25-round characteristic probability is upper-bounded by \( 2^{-100} \). Our algorithm shows that the optimal characteristic probability is \( 2^{-110} \) in 0.5 second. The number of rounds is not a problem since an optimal 64-round characteristic is computed in just 2 seconds. Note that PRESENT has 32 rounds.

The permutation used in SMALLPRESENT [15] (and in PRESENT) can be generalized for each integers \( N \) and \( S \). Define \( \phi_{S,N} \) for all \( 1 \leq k \leq SN \) by

\[
\phi_{S,N}(k) = N(k-1 \mod S) + \left\lfloor \frac{k-1}{S} \right\rfloor + 1.
\]

It is easy to verify that the permutation \( \phi \) used in Example 9 is \( \phi_{4,4} \). We have constructed a 128-bit SPN on the same model as PRESENT to test our algorithm efficiency. Define \( \pi \) as the bit permutation associated with \( \phi_{8,16} \) and the 16 S-boxes as the AES S-box. We have obtained an optimal 13-round characteristic of probability \( 2^{-89} \) in 7.1 seconds.

To analyze PUﬃn security against differential cryptanalysis, Cheng et al [13] have upper-bounded the probability of an optimal 31-round characteristic by \( 2^{-62} \). In 0.02 second, we have computed a characteristic that reaches this
bound. It is obtained by extending the following iterative characteristic:

$$\begin{align*}
\alpha_1 &= (4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
\beta_1 &= (4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
\alpha_{i+1} &= (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
\beta_{i+1} &= (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0).
\end{align*}$$

Finally, we have tested our algorithm on ICEBERG [14]. Its permutation layer is not a bit permutation but a linear diffusion. The optimization presented in Section IV-C is thus no longer applicable. The authors have upper-bound the probability of an optimal 16-rounds characteristic by $2^{-160}$. We proved that it is in fact $2^{-171.6}$ in 2.3 seconds. All these results are outlined in Figure 5.

**CONCLUSION**

In this paper, we have presented a generic algorithm that computes a maximum probability differential characteristic in a SPN. Running this algorithm may allow to prove the practical security of the block cipher. In the opposite case of weak cipher, the returned characteristic allows the cryptanalyst to build an optimal attack.

Especially optimized for SPN using a bit permutation as permutation layer, we are able to find a maximum probability characteristic of PRESENT and PUFFIN within one second. Block cipher designers have then a powerful tool which can be run several times to improve block cipher components.

**REFERENCES**


<table>
<thead>
<tr>
<th>Cipher</th>
<th>Block size</th>
<th>Round number</th>
<th>Upper-bound</th>
<th>Best probability</th>
<th>Running time</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRESENT</td>
<td>64</td>
<td>5</td>
<td>$2^{-20}$</td>
<td>$2^{-20}$</td>
<td>0.3 s</td>
</tr>
<tr>
<td>PRESENT</td>
<td>64</td>
<td>25</td>
<td>$2^{-100}$</td>
<td>$2^{-110}$</td>
<td>0.5 s</td>
</tr>
<tr>
<td>PRESENT-like</td>
<td>128</td>
<td>13</td>
<td>–</td>
<td>$2^{-89}$</td>
<td>7.1 s</td>
</tr>
<tr>
<td>PUFFIN</td>
<td>64</td>
<td>31</td>
<td>$2^{-62}$</td>
<td>$2^{-62}$</td>
<td>0.02 s</td>
</tr>
<tr>
<td>ICEBERG</td>
<td>64</td>
<td>16</td>
<td>$2^{-160}$</td>
<td>$2^{-171.6}$</td>
<td>2.3 s</td>
</tr>
</tbody>
</table>

Figure 5. Summary of Results