Krzysztof CHMIEL
POZNAŃ UNIVERSITY OF TECHNOLOGY, INSTITUTE OF CONTROL AND INFORMATION ENGINEERING
pl. Marii Skłodowskiej Curie 5, 60-965 Poznań

Approximation table computing algorithms in cryptanalysis of block ciphers

Ph.D. eng. Krzysztof CHMIEL

Assistant professor at Poznań University of Technology, Poland. His research and scientific interests focus on data security in information systems and cryptology, especially methods of designing and cryptanalysis of cryptographic algorithms. He is the author of a number of publications on differential and linear approximation of block ciphers and their component functions.

e-mail: krzysztof.chmiel@put.poznan.pl

Abstract

Approximation algorithms based on definitions of differential and linear equations, developed for computation of single element of the approximation tables, are of exponential time complexity. Fast general algorithms, for computation the best nonzero approximations in at worst linear time for a single element, without memory needed for storage of the whole table are presented in the paper. To frequently used components of block ciphers belong arithmetic sum and subtraction functions. For these functions are presented fast specialized algorithms computing a single element of the approximation tables in linear time.

Keywords: block cipher, cryptanalysis, differential approximation, linear approximation.

Algorytmy obliczania tablic aproksymacji w kryptoanalizie szyfrów blokowych

Streszczenie

Do najważniejszych ogólnych metod analizy kryptograficznej szyfrów blokowych należą kryptoanaliza różnicowa i kryptoanaliza liniowa. W obu metodach wykorzystywane są równania, które w sposób przybliżony, z pewnym prawdopodobieństwem, opisują działanie szyfru. Równania te nazywane są aproksymacjami różnicowymi lub liniowymi. Dla dowolnej funkcji f o n binarnych wejściach i m binarnych wyjściach zbiór wszystkich aproksymacji różnicowych lub liniowych może być reprezentowany w postaci tablicy aproksymacji o rozmiarze $O(2^{n+m})$. W artykule przedstawiono algorytmy obliczania tych tablic. Oparte na definicji aproksymacji różnicowej lub liniowej algorytmy obliczają pojedynczą wartość tablicy aproksymacji w czasie wykładniczym. Ogranicza to zastosowanie tych podstawowych algorytmów do funkcji składowych szyfru o niewielkiej liczbie binarnych wejść i wyjść. Przedstawione w artykule szybkie ogólne algorytmy obliczają najlepszą niezerową aproksymację różnicową i liniową w co najwyżej liniowym czasie $O(n+m)$ dla pojedynczego elementu bez angażowania pamięci potrzebnej do przechowania całych tablic. Do często stosowanych elementów składowych szyfrów blokowych należą funkcje sumy i różnicy arytmetycznej. Dla tych funkcji przedstawiono w artykule szybkie specjalizowane algorytmy obliczające pojedynczy element tablic aproksymacji w czasie liniowym.

Słowa kluczowe: szyfr blokowy, kryptoanaliza, aproksymacja różnicowa, aproksymacja liniowa.

1. Introduction

Differential and linear cryptanalysis belong to main topics in cryptology since they were introduced and successfully applied to the Data Encryption Standard. Unlike the differential cryptanalysis, which is essentially a chosen-plaintext attack [2, 17, 18, 21], the linear cryptanalysis is essentially a known-plaintext attack and moreover is applicable to an only-ciphertext attack under some circumstances [12-20, 22, 23, 38, 39].

Various improvements and extensions of the two methods have been proposed, such as: differential-linear cryptanalysis [6, 35], truncated differential cryptanalysis [33, 40], higher order differential cryptanalysis [29, 33], related key cryptanalysis [3, 7, 8, 10, 11, 26, 32], boomerang attack [7, 41], inside-out attack [41], impossible differential cryptanalysis [4, 27, 37], slide attack [9, 25], rectangle attack [5, 7, 26, 32], multiple-linear cryptanalysis [1, 30, 36], bi-linear cryptanalysis [24, 42], non-linear cryptanalysis [34], partitioning cryptanalysis [28], mod n cryptanalysis [31].

In the case of iterative block ciphers, the calculation of the most effective differential or linear approximations is carried out typically in two main steps. First, as a result of composition of approximations of component functions, the effective approximations of a single iteration are calculated. Next, as a result of composition of approximations of consecutive iterations, the approximation of the entire algorithm is obtained.

The basic idea of differential cryptanalysis is to analyze the effect of particular differences in plaintext pairs on the differences of the resultant ciphertext pairs. The differences are usually calculated as a result of XOR operation. Input XOR of a cipher algorithm causes a specified output XOR with some probability. The appropriate, approximate expression will be called the differential approximation.

By the *differential approximation* of function $y = f(x)$: $\{0,1\}^n \rightarrow \{0,1\}^m$ we mean an arbitrary equation of the form:

$$f(x) \oplus f(x \oplus x') = y', \tag{1}$$

for $x' \in \{0,1\}^n$, $y' \in \{0,1\}^m$, which is fulfilled with approximation probability

$$p = N(x',y') / 2^n, \tag{2}$$

where

$$N(x', y') = |\{x \in \{0,1\}^n : f(x) \oplus f(x \oplus x') = y'\}|. \tag{3}$$

The sequences x', y' are called input and output *difference* respectively, and the function $N(x',y')$ is called the *counting function* of the approximation. The magnitude of p represents the *effectiveness* of the approximation. Among approximations we distinguish the *zero differential approximation*, with $x' = 0^n \in \{0\}^n$ and $y' = 0^m \in \{0\}^m$, which probability p is equal to 1 for arbitrary function f. Differences x', y' are equivalently denoted by numbers $x' \in \{0, 1, ..., 2^n-1\}$ and $y' \in \{0, 1, ..., 2^m-1\}$.

The basic idea of linear cryptanalysis is to describe a given cipher algorithm by a linear approximate expression, so-called linear approximation. In general, the *linear approximation* of function $y = f(x)$: $\{0, 1\}^n \rightarrow \{0, 1\}^m$ is defined as an arbitrary equation of the form:

$$\bigoplus_{i \in y'} y_i = \bigoplus_{j \in x'} x_j, \tag{4}$$

or in the simplified notation:

$$y[y'] = x[x'], \tag{5}$$

for $x' \subseteq \{1, 2, ..., n\}$, $y' \subseteq \{1, 2, ..., m\}$, which is fulfilled with approximation probability

$$p = N(x',y') / 2^n, \tag{6}$$

where

$$N(x', y') = |\{x \in \{0,1\}^n : f(x)[y'] = x[x']\}|. \quad (7)$$

The sets of indexes x', y' are called input and output *mask* respectively, and the function $N(x',y')$ is called the *counting function* of the approximation. The *effectiveness* of the approximation is represented by magnitude of

$$|\Delta p| = |p - 1/2|. \quad (8)$$

By the *zero linear approximation* we mean approximation with $x' = y' = \Phi$, which probability p is equal to 1 for arbitrary function f. Masks x', y' are often denoted by numbers $x' \in \{0, 1, ..., 2^n-1\}$ and $y' \in \{0, 1, ..., 2^m-1\}$, corresponding to the zero-one representation of sets.

2. Approximation tables

The set of all differential approximations of function f can be described in the form of the *approximation table TDf*, called in [2] the *difference distribution table*. The element $TDf[x', y']$ of the table, is defined as follows:

$$TDf[x', y'] = N(x', y'). \quad (9)$$

The maximum value of *TDf*, that corresponds to the best, i.e. most effective, nonzero differential approximation, is denoted by *maxTD*, and is defined by formula:

$$maxTD = max\{TDf[x', y'] : x' \neq 0^n \vee y' \neq 0^m\}. \quad (10)$$

Similarly, the set of all linear approximations of function f is represented in the form of the *approximation table TAf*. The element $TAf[x', y']$ of the table, is defined as follows:

$$TAf[x', y'] = \Delta N(x', y') = N(x', y') - 2^{n-1}. \quad (11)$$

The maximum absolute value of *TAf*, that corresponds to the best nonzero linear approximation, is denoted by *maxTA* and is defined in the following way:

$$maxTA = max\{|TAf[x', y']| : x' \neq \Phi \vee y' \neq \Phi\}. \quad (12)$$

Tab. 1. An example function f and its approximation tables *TDf* and *TAf* (n=4, m=2)
Tab. 1. Przykładowa funkcja f i jej tablice aproksymacji *TDf* oraz *TAf* (n=4, m=2)

x	$y=f(x)$
0	3
1	3
2	3
3	0
4	1
5	3
6	1
7	1
8	0
9	0
10	3
11	3
12	1
13	2
14	2
15	2

(f)

x'	y'			
	0	1	2	3
0	16	0	0	0
1	10	0	2	4
2	6	0	2	8
3	6	0	2	8
4	2	8	6	0
5	2	8	6	0
6	0	2	12	2
7	2	4	10	0
8	4	2	0	10
9	2	0	2	12
10	8	2	0	6
11	8	2	0	6
12	0	6	8	2
13	0	6	8	2
14	2	8	6	0
15	2	12	2	0

(*TDf*)

x'	y'			
	0	1	2	3
0	8	–2	–1	1
1	0	–2	1	–1
2	0	0	1	1
3	0	0	3	–1
4	0	0	–1	7
5	0	0	–3	1
6	0	2	1	–1
7	0	2	–1	1
8	0	–4	1	1
9	0	0	–1	–1
10	0	–2	–5	1
11	0	2	1	–1
12	0	2	–3	–1
13	0	–2	–1	1
14	0	–4	–1	–1
15	0	0	1	1

(*TAf*)

The approximation tables of an example function f are presented in table 1. There exist many effective approximations of the function, identified by nonzero values of the tables. The best nonzero differential approximations have *maxTD* = 12 and probability p = 12/16, while the best nonzero linear approximation has *maxTA* = 7, and effectiveness $|\Delta p|$ = 7/16.

3. Basic algorithms

The basic algorithm, computing a single element of the approximation table *TDf*, is presented in Fig. 1. Function N(...) is the counting function of the approximation. The main function TD-F(...) returns the value of this function. The time complexity of the algorithm is $O(2^n)$.

```
TD-F(x', y', f, n, m)
 1. N(x', y', f, n, m)
 2.  w ← 0
 3.    for x ← 0 to 2^n − 1 do
 4.       if f(x) ⊕ f(x ⊕ x') = y'
 5.       then w ← w + 1
 6.    return w
 7. return N(x', y', f, n, m)
```

Fig. 1. Basic algorithm computing a single element of the approximation table *TDf*
Rys. 1. Podstawowy algorytm obliczania pojedynczego elementu tablicy aproksymacji *TDf*

In Fig. 2 the basic algorithm computing a single element of the approximation table *TAf* is presented. Auxiliary function BIT-XOR(...) computes the XOR of the n least significant bits of parameter x. Function N(...) is the counting function of the approximation. The main function TA-F(...) returns the value of ΔN. The time complexity of the algorithm is $O((n+m)\cdot 2^n)$.

```
TA-F(x', y', f, n, m)
 1. BIT-XOR(x, n)
 2.    w ← 0
 3.    for i ← 0 to n − 1 do w ← w ⊕ x_i
 4.    return w
 5. N(x', y', f, n, m)
 6.    w ← 0
 7.    for x ← 0 to 2^n − 1 do
 8.       y ← f(x)
 9.       if BIT-XOR(x and x', n) = BIT-XOR(y and y', m)
10.       then w ← w + 1
11.    return w
12. return N(x', y', f, n, m) − 2^(n−1)
```

Fig. 2. Basic algorithm computing a single element of the approximation table *TAf*
Rys. 2. Podstawowy algorytm obliczania pojedynczego elementu tablicy aproksymacji *TAf*

The size of the approximation tables *TDf* and *TAf* of function f is equal to 2^{n+m}, and the basic algorithms compute a single element of the tables in exponential time. The presented in the next chapter fast algorithms compute the approximation tables in at worst linear time for a single element.

4. Fast general algorithms

The fast general computation of approximation table *TDf*, for an arbitrary function $y = f(x)$: $\{0,1\}^n \rightarrow \{0,1\}^m$, where n, $m \geq 1$, is suggested in [2]. Each row of *TDf* contains in fact the distribution of output differences y' for all input pairs with difference x'. By examination of all input pairs $(x, x \oplus x')$, the whole row of *TDf* can

be computed instead of a single element. The appropriate procedure is presented in Fig. 3.

```
CALC-TDR(TDR, x', f, n, m)
1. for y' ← 0 to 2^m–1 do TDR[y'] ← 0
2. for x ← 0 to 2^n–1 do
3.   y' ← f(x) ⊕ f(x ⊕ x')
4.   TDR[y'] ← TDR[y'] +1
5. return
```

Fig. 3. Computing procedure of the row of the approximation table *TDf* for *x'*
Rys. 3. Procedura obliczania wiersza tablicy aproksymacji *TDf* dla *x'*

Procedure CALC-TDR(...) computes the row of *TDf* for *x'* in time $O(2^{n-m})$ for a single element. If $n - m$ is limited by a constant, in particular for $n = m$, then the computation time is O(1) for a single element.

```
maxTD(f, n, m)
1. max ← 0
2. for x' ← 0 to 2^n–1 do
3.   CALC-TDR(TDR, x', f, n, m)
4.   for y' ← 0 to 2^m–1 do
5.     if x' ≠ 0 or y' ≠ 0 then
6.       if max < TDR[y'] then max ← TDR[y']
7. return max
```

Fig. 4. Fast algorithm computing the value of *maxTD* for function *f*
Rys. 4. Szybki algorytm obliczania wartości *maxTD* dla funkcji *f*

The fast algorithm maxTD(...) presented in Fig. 4, computes the value of *maxTD*, that corresponds to the best nonzero differential approximation of function *f*. The computation is carried out row by row of *TDf*, in time O(1) for a single element if $n - m$ is limited by a constant.

The fast general algorithm computing approximation table *TAf*, first described in [16], is composed of two main steps. In the first step the initial value of *TAf* is computed. The initial *TAf* contains elementary approximation tables of all residual functions of *f*, dependent on one variable x_0. In the second step the final value of *TAf* is computed, as a result of addition and subtraction of these elementary tables for consecutive variables. An important feature of the fast algorithm is, that the computation of *TAf* can be carried out column by column, without keeping the whole *TAf* in memory.

```
INI-TAC(TAC, y', f, n, m, TP)
1. for x' ← 0 to 2^n–1 do
2.   TAC[x'] ← BIT-XOR(f(x') and y', m)
3. for x' ← 0 to 2^n–2 step 2 do
4.   (TAC[x'], TAC[x'+1]) ← TP[TAC[x'], TAC[x'+1]]
5. return
```

Fig. 5. Procedure computing the initial value of *TAf* column for *y'*
Rys. 5. Procedura obliczania początkowej wartości kolumny *TAf* dla *y'*

The initial value of *TAf* column for *y'* is computed by procedure INI-TAC(...) presented in Fig. 5. First, in steps 1-2, for each mask *x'*, is calculated the value *y*[*y'*] with use of the auxiliary function BIT-XOR(...) from Fig. 2. Then, in steps 3-4, each pair of adjacent values, corresponding to the value 0 and 1 of variable x_0, is replaced by a pair stored in so called table of pairs *TP*.

Table *TP* of pairs is presented in table 2. For each function of one variable, defined by the values of v_0 and v_1, it contains a pair of values from the right column of the appropriate elementary approximation table.

Tab. 2. Table *TP* of pairs
Tab. 2. Tablica par *TP*

v_0	v_1	$TP[v_0, v_1]$
0	**0**	(1, 0)
0	**1**	(0, 1)
1	**0**	(0,–1)
1	**1**	(–1, 0)

```
CALC-TAC(TAC, i, j)
1. if j – i > 2 then
2.   k ← (i + j) div 2
3.   CALC-TAC(TAC, i, k)
4.   CALC-TAC(TAC, k+1, j)
5.   SUMSUB-TAC(TAC, i, k, k+1, j)
6. return
```

Fig. 6. Procedure computing the final value of *TAf* column for *y'*, first call: CALC-TAC(*TAC*, 0, 2^n–1)
Rys. 6. Procedura obliczania końcowej wartości kolumny *TAf* dla *y'*, pierwsze wywołanie: CALC-TAC(*TAC*, 0, 2^n–1)

The final value of *TAf* column for *y'* is computed by the recursive procedure CALC-TAC(...) presented in Fig. 6. In the first call of the procedure the initial value of *TAf* column must be used and the range of rows is from $i = 0$ to $j = 2^n–1$. For the range greater than 2, the problem is solved by solution of two half-subproblems. Having solved the subproblems, the approximation table column of the problem is computed by the auxiliary procedure SUMSUB-TAC(...) from Fig. 7.

```
SUMSUB-TAC(TAC, i1, j1, i2, j2)
1. for i ← 0 to j1 – i1 do
2.   (TAC[i1+i], TAC[i2+i]) ← (TAC[i1+i] + TAC[i2+i],
3.                            TAC[i1+i] – TAC[i2+i])
4. return
```

Fig. 7. Procedure SUMSUB-TAC(...)
Rys. 7. Procedura SUMSUB-TAC(...)

Procedure SUMSUB-TAC(...), for two parts of *TAf* column that correspond to the subproblems, replaces first of them by their sum and the second by their difference. It can be shown, that procedures INI-TAC(...) and CALC-TAC(...) compute the column of *TAf* for *y'* in linear time $O(n + m)$ for a single element [16].

```
maxTA(f, n, m)
1. max ← 0
2. for y' ← 0 to 2^m–1 do
3.   INI-TAC(TAC, y', f, n, m, TP)
4.   CALC-TAC(TAC, 0, 2^n–1)
5.   for x' ← 0 to 2^n–1 do
6.     if x' ≠ 0 or y' ≠ 0 then
7.       if max < | TAC[x'] | then max ← | TAC[x'] |
8. return max
```

Fig. 8. Fast algorithm computing the value of *maxTA* for function *f*
Rys. 8. Szybki algorytm obliczania wartości *maxTA* dla funkcji *f*

The fast algorithm maxTA(...) presented in Fig. 8, computes the value of *maxTA*, that corresponds to the best nonzero linear approximation of function *f*. The computation is carried out column by column of *TAf*, in time $O(n + m)$ for a single element.

5. Fast specialized algorithms

For some functions like *n*-bit arithmetic sum function $z = SUMn(x, y) = (x + y) \bmod 2^n$, there exist specialized algorithms that compute a single element of the approximation

tables *TDSUMn* and *TASUMn*, in linear time. The approximation tables of 2-bit function *SUM*2 are presented in table 3. In *TDSUM*2 there exist four approximations with probability $p = 1$ and twenty four approximations with probability $p = 1/2$. In *TASUM*2 there exist two approximations with $\Delta p = 1/2$ and eight approximations with effectiveness $|\Delta p| = 1/4$.

Tab. 3. Function *SUM*2 and its approximation tables *TDSUM*2 and *TASUM*2
Tab. 3. Funkcja *SUM*2 i jej tablice aproksymacji *TDSUM*2 oraz *TASUM*2

x	y	z
0	0	0
0	1	1
0	2	2
0	3	3
1	0	1
1	1	2
1	2	3
1	3	0
2	0	2
2	1	3
2	2	0
2	3	1
3	0	3
3	1	0
3	2	1
3	3	2

(*SUM*2)

x'	y'	z'			
		0	1	2	3
0	0	16	0	0	0
0	1	0	8	0	8
0	2	0	0	16	0
0	3	0	8	0	8
1	0	0	8	0	8
1	1	8	0	8	0
1	2	0	8	0	8
1	3	8	0	8	0
2	0	0	0	16	0
2	1	0	8	0	8
2	2	16	0	0	0
2	3	0	8	0	8
3	0	0	8	0	8
3	1	8	0	8	0
3	2	0	8	0	8
3	3	8	0	8	0

(*TDSUM*2)

x'	y'	z'			
		0	1	2	3
0	0	8	0	0	0
0	1	0	0	0	0
0	2	0	0	0	0
0	3	0	0	0	0
1	0	0	0	0	0
1	1	0	8	0	0
1	2	0	0	0	0
1	3	0	0	0	0
2	0	0	0	0	0
2	1	0	0	0	0
2	2	0	0	4	–4
2	3	0	0	4	4
3	0	0	0	0	0
3	1	0	0	0	0
3	2	0	0	4	4
3	3	0	0	–4	4

(*TASUM*2)

Function *SUMn* is treated as composed of n identical 1-bit cells, containing the carry function f. The approximation tables *TDC* and *TAC* of carry function f are shown in table 4. In *TDC* there exist two approximations with probability $p_i = 1$ and twelve approximations with probability $p_i = 1/2$. In *TAC*, besides the zero-approximation with $\Delta p = 1/2$, there exist four approximations with effectiveness $|\Delta p| = 1/4$.

Tab. 4. Carry function f and its approximation tables *TDC* and *TAC*
Tab. 4. Funkcja przeniesienia f i jej tablice aproksymacji *TDC* oraz *TAC*

x_{i-1}	y_{i-1}	c_{i-1}	c_i
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

(f)

x_{i-1}'	y_{i-1}'	c_{i-1}'	c_i'	
			0	1
0	0	0	8	0
0	0	1	4	4
0	1	0	4	4
0	1	1	4	4
1	0	0	4	4
1	0	1	4	4
1	1	0	4	4
1	1	1	0	8

(*TDC*)

$x_{i-1}' \oplus z_{i-1}'$	$y_{i-1}' \oplus z_{i-1}'$	$c_{i-1}' \oplus z_{i-1}'$	c_i'	
			0	1
0	0	0	4	0
0	0	1	0	2
0	1	0	0	2
0	1	1	0	0
1	0	0	0	2
1	0	1	0	0
1	1	0	0	0
1	1	1	0	-2

(*TAC*)

The algorithm computing the value of the approximation table *TDSUMn* for differences x', y', z' is presented in Fig. 9.

TD-SUMn(TDC, x', y', z', n)
1. TD-SUMnR(TDC, x', y', z', i, c')
2. $c_{i-1}' \leftarrow x_{i-1}' \oplus y_{i-1}' \oplus z_{i-1}'$
3. $w_i \leftarrow TDC[\, x_{i-1}' , y_{i-1}' , c_{i-1}' , c_i']$
4. **if** $i > 1$
5. **then return** $w_i \cdot$ TD-SUMnR(TDC, x', y', z', i-1, c')
6. **else if** $c_{i-1}' = 0$ **then return** w_i **else return** 0
7. $c_{n-1}' \leftarrow x_{n-1}' \oplus y_{n-1}' \oplus z_{n-1}'$
8. **if** $n > 1$
9. **then return** $(1/2^{n-3}) \cdot$TD-SUMnR(TDC, x', y', z', n-1,c')
10. **else if** $c_{n-1}' = 0$ **then return** 4 **else return** 0

Fig. 9. Fast specialized algorithm computing a value of *TDSUMn*
Rys. 9. Szybki specjalizowany algorytm obliczania wartości *TDSUMn*

Argument *TDC* denotes the approximation table of carry function f. In the recursive function TD-SUMnR(...), argument i denotes the current cell number and argument c' is used to store successively computed bits of carry difference. The time complexity of the algorithm is $O(n)$, which is much better in comparison to complexity $O(2^{2n})$ of the basic algorithm for function *SUMn*.

In Fig. 10 a recursive algorithm, computing the values of the linear approximation table of function *SUMn* for masks x', y', z' is presented. Argument *TAC* denotes the approximation table of carry function f and argument i denotes the current cell number. Argument c', in which additionally successively computed bits of carry mask are stored, is used to input the value of output carry mask c_i' of cell i. In the first call of the function, TA-SUMn(*TAC*, $x', y', z', n, 0$), it is necessary to fulfil the condition $c_n' = 0$.

TA-SUMn(TAC, x', y', z', i, c')
1. $c_{i-1}' \leftarrow c_i' \oplus x_{i-1}' \oplus y_{i-1}' \oplus z_{i-1}'$
2. $w_i \leftarrow TAC[\, x_{i-1}' \oplus z_{i-1}' , y_{i-1}' \oplus z_{i-1}' , c_{i-1}' \oplus z_{i-1}' , c_i']$
3. **if** $i > 1$
4. **then return** $w_i \cdot$ TA-SUMn(TAC, x', y', z', i-1, c')
5. **else return** $w_i \cdot 1/2$

Fig. 10. Fast specialized algorithm computing a value of *TASUMn*
Rys. 10. Szybki specjalizowany algorytm obliczania wartości *TASUMn*

The complexity of the algorithm from Fig. 10 is $O(n)$ which is much better in comparison to the complexity $O(n \cdot 2^{2n})$ of the basic algorithm for function *SUMn*.

6. Conclusion

The basic algorithms computing a single element of the approximation tables *TDf* and *TAf* are of exponential time complexity. The presented in the paper fast general algorithms compute the values of *maxTD* and *maxTA* in at worst linear time for a single element, without memory needed for storage of the whole table. For some functions like n-bit arithmetic sum function or subtraction function, there exist specialized fast algorithms, that compute a single value of the approximation tables in linear time.

The work was supported by finances of West Pomeranian Provincial Administration.

This work was partially supported by the Polish Ministry of Education and Science as a 2010–2013 research project.

7. References

[1] Benoit G., Tillich J.-P.: On Linear Cryptanalysis with Many Linear Approximations. http://eprint.iacr.org/2009/463.
[2] Biham E., Shamir A.: Differential Cryptanalysis of the Data Encryption Standard, Springer-Verlag, Berlin Heidelberg New York 1993.
[3] Biham E.: New Types of Cryptanalytic Attacks Using Related Keys. EUROCRYPT'93, LNCS 765, 398-409, Springer 1994.
[4] Biham E., Biryukov A., Shamir A.: Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials, EUROCRYPT'99, LNCS 1592, 12-23, Springer 1999.
[5] Biham E., Dunkelman O., Keller N.: The Rectangle Attack - Rectangling the Serpent. EUROCRYPT 2001, LNCS 2045, 340-357, Springer 2001.
[6] Biham E., Dunkelman O., Keller N.: Differential-Linear Cryptanalysis of Serpent. FSE 2003, LNCS 2887, 9-21, Springer 2003.
[7] Biham E., Dunkelman O., Keller N.: Related-Key Boomerang and Rectangle Attacks. EUROCRYPT 2005, LNCS 3494, 507-525, Springer 2005.
[8] Biham E., Dunkelman O., Keller N.: A Unified Approach to Related-Key Attacks, FSE 2008, LNCS 5086, 73-96, Springer 2008.
[9] Biryukov A., Wagner D.: Slide Attacks. FSE'99, LNCS 1636, 245–259, Springer 1999.

[10] Biryukov A., Khovratovich D., Nikolic I.: Distinguisher and Related-Key Attack on the Full AES-256. CRYPTO 2009, LNCS 5677, 231–249 , Springer 2009.
[11] Biryukov A., Khovratovich D.: Related-key Cryptanalysis of the Full AES-192 and AES-256. ASIACRYPT 2009, LNCS 5912, 1–18, Springer 2009.
[12] Bucholc K., Chmiel K., Grocholewska-Czuryło A., Idzikowska E., Janicka-Lipska I., Stokłosa J.: Scalable PP-1 Block Cipher, International Journal of Applied Mathematics and Computer Science, Vol. 20, No 2 (2010), 401-411.
[13] Chmiel K.: Linear Approximation of Arithmetic Sum Function. In Sołdek J., Drobiazgiewicz L. (Eds): Artificial Intelligence and Security in Computing Systems, Kluwer Academic Publishers, 293-302, Boston/Dordrecht/London 2003.
[14] Chmiel K.: Linear Approximation of Structures with Selectors, Proceedings of 6-th NATO Regional Conference on Military Communications and Information Systems 2004, WIŁ, 269-273, Zegrze 2004.
[15] Chmiel K.: On Arithmetic Subtraction Linear Approximation. In Pejaś J., Piegat A. (Eds): Enhanced Methods in Computer Security, Biometric and Artificial Intelligence Systems, Kluwer Academic Publishers, 125-134, New York 2005.
[16] Chmiel K.: Fast Computation of Approximation Tables. In Saeed K., Pejaś, J. (Eds): Information Processing and Security Systems, Springer Verlag, 125-134, Berlin Heidelberg New York 2005.
[17] Chmiel K.: On Differential and Linear Approximation of S-box Functions. In Saeed K., Pejaś, J., Mosdorf R. (Eds): Biometrics, Computer Security Systems and Artificial Intelligence Applications, Springer, 111-120, New York 2006.
[18] Chmiel K.: Distribution of the Best Nonzero Differential and Linear Approximations of S-box Functions. Journal of Telecommunications and Information Technology, vol. 3, 2006, 8-13.
[19] Chmiel K.: Intermediate Evaluation of DES-like Cryptosystems. Proceedings of Military CIS Conference, MCC 2007, (Bonn, Sept. 25-26), ISBN 978-3-934401-16-7, 1-7, Bonn 2007.
[20] Chmiel K.: On intermediate evaluation of block ciphers. In Pejaś J., Saeed K. (Eds): Advances in Information Processing and Protection, Springer, 251-261, New York 2007.
[21] Chmiel K.: Differential Approximation of Arithmetic Sum Function. Polish Journal of Environmental Studies, Vol. 16, No 5B (2007), 299-303.
[22] Chmiel K., Grocholewska-Czuryło A., Stokłosa J.: Involutional Block Cipher for Limited Resources. Proceedings of IEEE GLOBECOM Conference, 1852-1856, New Orleans 2008.
[23] Chmiel K.: Rough Evaluation of Block Ciphers. Measurements, Automation and Monitoring (PAK), vol. 55, nr 10, 2009, 835-838.
[24] Courtois N.T.: Feistel Schemes and Bi-linear Cryptanalysis. CRYPTO 2004, LNCS 3152, 23-40, Springer 2004.
[25] Courtois N.T., Bard G.V., Wagner D.: Algebraic and Slide Attacks on KeeLoq, FSE 2008, LNCS 5086, 97-115, Springer 2008.
[26] Dunkelman O., Keller N., Kim J.: Related-Key Rectangle Attack on the Full SHACAL-1, SAC 2006, LNCS 4356, 28-44, Springer 2007.
[27] Dunkelman O., Keller N.: An Improved Impossible Differential Attack on MISTY1. ASIACRYPT 2008, LNCS 5350, 441-454, Springer 2008.
[28] Harpes C., Massey J.: Partitioning Cryptanalysis, FSE'97, LNCS 1267, 13–27. Springer 1997.
[29] Hatano Y., Sekine H., Kaneko T.: Higher Order Differential Attack of Camellia(II). SAC 2002, LNCS 2595, 129-146, Springer 2003.
[30] Kaliski Jr. B.S., Robshaw M.J.B.: Linear Cryptanalysis Using Multiple Approximations and FEAL, FSE'94, LNCS 1008, 249–264, Springer 1995.
[31] Kelsey J., Schneier B., Wagner D.: Mod n Cryptanalysis, with Applications Against RC5P and M6. FSE'99, LNCS 1636, 139–155, Springer 1999.
[32] Kim J., Hong S., Preneel B.: Related-Key Rectangle Attacks on Reduced AES-192 and AES-256. FSE 2007, LNCS 4593, 225-241, Springer 2007.
[33] Knudsen L.: Truncated and Higher Order Differential, FSE'94, LNCS 1008, 196-211, Springer 1995.
[34] Knudsen L.R., Robshaw M.J.B.: Non-linear Approximations in Linear Cryptanalysis. EUROCRYPT'96, LNCS 1070, 224–236, Springer 1996.
[35] Langford S., Hellman M.: Differential-Linear Cryptanalysis, CRYPTO'94, LNCS 839, 17-25, Springer 1994.
[36] Liu Z., Gu D., Zhang J.: Multiple Linear Cryptanalysis of Reduced-Round SMS4 Block Cipher. http://eprint.iacr.org/2009/256.
[37] Lu J., Dunkelman O., Keller N., Kim J.: New Impossible Differential Attacks on AES. http://eprint.iacr.org/2008/540.
[38] Matsui M.: Linear Cryptanalysis Method for DES Cipher, EUROCRYPT'93, LNCS 765, 386–397, Springer 1994.
[39] Matsui M.: The First Experimental Cryptanalysis of the Data Encryption Standard. CRYPTO'94, LNCS 839, 1–11, Springer 1994.
[40] Reichardt B.W., Wagner D.: Markov Truncated Differential Cryptanalysis of Skipjack. SAC 2002, LNCS 2595, 110-128, Springer 2003.
[41] Wagner D.: A Boomerang Attack. FSE'99, LNCS 1636, 156-170, Springer 1999.
[42] Zhang H., Wang S., Wang X.: The Probability Advantages of Two Linear Expressions in Symmetric Ciphers, http://eprint.iacr.org/2006/242.

otrzymano / received: 10.07.2010
przyjęto do druku / accepted: 01.09.2010 *artykuł recenzowany*

INFORMACJE

Nowy dział „Niepewność wyników pomiarów” na stronie internetowej Wydawnictwa PAK

Uprzejmie informuję, że na stronie internetowej Wydawnictwa PAK (WWW.pak.info.pl) został utworzony dział „Niepewność wyników pomiarów”. Na p.o. redaktora działu został powołany dr inż. Paweł Fotowicz.

Dr P. Fotowicz jest ekspertem w zakresie problematyki niepewności, autorem szeregu wartościowych publikacji w czasopismach krajowych i zagranicznych. Prezentował swoje prace na licznych konferencjach i warsztatach szkoleniowych.

W dziale „ Niepewność wyników pomiarów”, obok dostępu do aktualnych wybranych opracowań dotyczących niepewności jest możliwość zadawania „Pytań do eksperta”. Pytania powinny być konkretne i szczegółowo sprecyzowane.

Pytania i odpowiedzi o istotnym znaczeniu dla szerszego grona metrologów będą archiwizowane i dostępne dla użytkowników strony internetowej Wydawnictwa PAK.

Zapraszam do odwiedzania działu „Niepewność wyników pomiarów” i do udziału w jego rozwoju.

Tadeusz SKUBIS
Redaktor naczelny Wydawnictwa PAK