THE DESIGN OF BOOLEAN FUNCTIONS BY MODIFIED HILL CLIMBING METHOD

Yuriy Izbenko[1], Vladislav Kovtun[2] and Alexandr Kuznetsov[3]

Abstract. With cryptographic investigations, the design of Boolean functions is a wide area. The Boolean functions play important role in the construction of a symmetric cryptosystem. In this paper the modified hill climbing method is considered. The method allows using hill climbing techniques to modify bent functions used to design balanced, highly nonlinear Boolean functions with high algebraic degree and low autocorrelation. The experimental results of constructing the cryptographically strong Boolean functions are presented.

When designing block and stream ciphers, Boolean functions play an important role and define the cryptographic strength of applications to differential and linear cryptanalysis particularly. Often the resistance of cryptosystems to known types of attacks is discussed in terms of Boolean functions used in them. A lot of attention has been given to construction of Boolean functions with desired cryptographic properties in cryptology [1–6]. The main strength criteria of Boolean functions are balancedness, high nonlinearity, high algebraic degree and low autocorrelation. There are three types of methods of constructing nonlinear Boolean functions: random generation, algebraic (or direct) and heuristic methods. Each of them has its own advantages and drawbacks.

Generating nonlinear Boolean function via random generation is too difficult to find functions that possess high cryptographic

[1] Nrjetix company. email: yuriy.izbenko@nrjetix.com
[2] Nrjetix company. email: vladislav.kovtun@nrjetix.com
[3] Nrjetix company. email: alexandr.kuznetsov@nrjetix.com

properties due to the vast size of search space, especially for function with $n > 8$, where n is the space size. The attractiveness of these techniques comes out of the simplicity in their implementation. Algebraic methods allow constructing functions that have a set of desired cryptographic properties with low computation complexity, but these functions can have low algebraic complexity [4]. The heuristic methods [1–6] are the newest techniques capable of effective Boolean functions generation with desired cryptographic properties. Because of some intuitive approaches used in heuristic methods and the fact that heuristic methods are not limited by algebraic constructions, these methods can construct Boolean functions with properties that are close to the maximum attained.

The core of all heuristic methods is the hill climbing method (HC) introduced in [1]. The HC method allows increasing the nonlinearity of a Boolean function, particularly of the randomly generated one. The HC method may be effectively used with genetic and simulated annealing methods. In this paper we consider a modification of the HC method, which allows constructing highly nonlinear Boolean functions with low autocorrelation. The main idea of the proposed method is 'inverting' of the HC's algorithm. There are two main differences in our method from HC method: 1) we are using a *bent function* as input data instead of a *randomly generated Boolean function*, 2) we are *decreasing* nonlinearity of the bent function to a required value instead of *increasing* nonlinearity of a randomly generated Boolean function.

This paper is structured as follows: Section 2 presents the main definitions and terms, Section 3 describes the modified hill climbing method, Section 4 shows the main results. In the final part we make conclusions of our investigations.

1. Preliminaries

An n-variable Boolean function is a function $f(x)$ with n input variables where the domain is the vector space $\mathbb{F}_2^n$ of binary input n-tuples $x = (x_1, x_2, \ldots, x_n)$ and the range is $\mathbb{F}_2$.

Boolean functions can be presented in one of the three well-known representations, namely in the binary truth table, the algebraic normal form and the Walsh - Hadamard representation. There is one-to-one correspondence between these representations of a Boolean function, and they each reflect a different aspect of the properties required for cryptography.

Definition 1.1. The *binary truth table* for a Boolean function on $\mathbb{F}_2^n$is $(0,1)$-sequence defined by $(f(\alpha_1), f(\alpha_2), \ldots, f(\alpha_{2^n-1}))$, and the sequence of $f(x)$ is $(1,-1)$-sequence defined by

$$((-1)^{f(\alpha_0)}, (-1)^{f(\alpha_1)}, \ldots, (-1)^{f(\alpha_{2^n-1})}),$$

where

$$\alpha_0 = (0,0,\ldots,0), \alpha_1 = (0,0,\ldots,1), \ldots, \alpha_{2^n-1} = (1,1,\ldots,1).$$

Definition 1.2. An n-variable Boolean function $f(x)$ is *balanced* if the output in the binary truth table contains an equal number of 0's and 1's (1's and -1's).

Definition 1.3. The *Algebraic Normal Form* (ANF) of a Boolean function $f(x)$ is

$$\begin{aligned} f(x_0, x_1, \ldots, x_n) &= \alpha_0 + \alpha_1 x_1 + \cdots + \alpha_n x_n + \\ &+ \alpha_{12} x_1 x_2 + \alpha_{13} x_1 x_3 + \cdots + \alpha_{12\ldots n} x_1 x_2 \ldots x_n \end{aligned} \tag{1}$$

where addition and multiplication are in $\mathbb{F}_2$.

Definition 1.4. The *algebraic degree* of a Boolean function $f(x)$, denoted by $\deg(f(x))$, is defined to be the maximum degree appearing in the ANF. Algebraic degree is an important property for Boolean functions and defines the linear complexity.

Definition 1.5. The Walsh-Hadamard transform (WHT) of Boolean function $f(x)$ is defined to be the real valued function $\mathbf{F}(w)$ over the vector space $\mathbb{F}_2^n$given by

$$\mathbf{F}(w) = \sum_x f(x)(-1)^{w \cdot x}, \tag{2}$$

where a dot product of vectors x and w is defined as $x \cdot w = x_1 w_1 + \cdots + x_n w_n$. WHT is important tool for the analysis of Boolean functions.

Definition 1.6. The *nonlinearity* of a Boolean function $f(x)$, denoted by $\mathbf{nl}(f(x))$, is

$$\mathbf{nl}(f(x)) = \min_{g \in A_n} d_H(f, g) \tag{3}$$

and defines Hamming distance to the nearest affine function. Here f and g are the binary truth table of $f(x)$ and $g(x)$, A_n is the set of affine functions on n variables and $d_H(f, g)$ is the Hamming distance between two vectors f and g, i.e., the number positions where f and g differ. Alternatively, the nonlinearity of $f(x)$ can be written in terms of Walsh-Hadamard transform as follows

$$\mathbf{nl}(f) = 2^{n-1} - \min |\mathbf{F}(w)|, w \neq 0. \tag{4}$$

Finding Boolean functions with maximal nonlinearity is an important and well studied problem. It is well known that only *bent functions* have maximal nonlinearity [7]. But firstly, bent functions are not balanced, and, secondly, they exist only if n is even. Thus, there is an open problem in finding the maximal nonlinearity for balanced functions for both n is even and n is odd.

Definition 1.7. The autocorrelation AC_f of Boolean function $f(x)$ is given by

$$AC_f = \max_s |\sum_x f(x) \cdot f(x \oplus s)|. \tag{5}$$

Good cryptographic functions have small AC_f. Only bent functions have $AC_f = 0$ for all $s \neq 0$, so called $AC_{\max}$. Maximal values of the AC_f are serious weakness, called the *linear structure*.

1.1. Nonlinearity of Boolean Functions

Nonlinearity is a crucial criterion of the strength of the Boolean functions. For a given n only bent functions f_{bent} have the maximal attainable nonlinearity [7]

$$\mathbf{nl}(f_{\text{bent}}) = 2^{n-1} - 2^{\frac{n}{2}-1}, \tag{6}$$

but bent functions are not balanced and exist only when n is even. An important task is to design a highly nonlinear balanced Boolean function. It is known that maximal attainable nonlinearity for Boolean functions is [8]:

$$\mathbf{nl}(f) \leq 2^{n-1} - 2^{\frac{n}{2}-1} - 2, \text{if } n \text{ is even} \tag{7}$$

$$\mathbf{nl}(f) \leq \lfloor 2^{n-1} - 2^{\frac{n}{2}-1} \rfloor, \text{if } n \text{ is odd.} \tag{8}$$

TABLE 1. Comparing the Nonlinearity of Balanced Functions

Method	Space dimension, n				
	4	6	8	10	12
Lowest Upper Bound, $\mathbf{nl}(f)_{\max}$	4	26	118	494	2014
Best known Example	4	26	116	492	2010
Bent Concatenation	4	24	112	480	1984
Our MHC	4	26	116	488	2002

However, the maximum nonlinearity attainable by balanced functions is not known. A lot of work discuss this problem, and attainable nonlinearity is still an open problem [3]. The highest nonlinearity found by our method is presented below in Table 1 for comparison with the lowest known upper bounds given by theory, and the best known examples [3]. One of the main open problems for practical applications is the maximal nonlinearity of balanced Boolean function for $n = 8$ is 116 or 118. We failed to construct the balanced function with $\mathbf{nl}(f) = 118$. However, our results show that we have attained the best known results in the class of heuristic methods.

2. Modified Hill Climbing method

As pointed in [1, 4], the hill climbing approach to Boolean function design is a means of improving the nonlinearity of a given Boolean function by making well chosen alterations of one or two places of the truth table. It has been shown that any single truth table change causes $\triangle_{\mathrm{WHT}} \in \{-2, 2\}$ for all w. Any two changes cause $\triangle_{\mathrm{WHT}} \in \{-4, 0, 4\}$. When the two function values satisfy $f(x_1) \neq f(x_2)$ then Hamming weight will not change. By starting with a balanced function, authors could hill climb to a more nonlinear balanced function by the method presented in [1]. That approach did not make an alteration to the truth table unless the nonlinearity was improved by such change. In this paper, we slightly modified the hill climbing method but have not introduced an alteration to the truth table of a bent sequence unless the nonlinearity was worse because of that change. As result of iterative operations, we can attain a high nonlinearity and low autocorrelation for a balanced Boolean function.

TABLE 2. Attained nonlinearity $\mathbf{nl}(f)_{\min}$

Method	Space dimension, n				
	4	6	8	10	12
Upper Bound of Nonlinearity, $\mathbf{nl}(f)_{\max}$	5	28	120	496	2016
Best known Example	4	26	116	492	2010
Needed position to change	2	4	8	16	32
Attained Nonlinearity, $\mathbf{nl}(f)_{\min}$	4	24	112	470	1984

It is well known that bent functions possess good cryptographic properties such as the highest nonlinearity and lowest autocorrelation. However, bent sequences are not balanced. The bent sequence with length 2^n has 2^{n-1} ones and $2^{n-1} + 2^{\frac{n}{2}-1}$ zeroes, or vice versa. The complementation of $2^{\frac{n}{2}-1}$ ones (zeroes) in bent sequence allows obtaining a balanced sequence with nonlinearity at least [7]

$$\mathbf{nl}(f) \geq 2^{n-1} - 2^{\frac{n}{2}}. \tag{9}$$

That property is an important tool applied in our method to save from unbalanced and attainable high nonlinearity of the modified bent sequence.

The main idea of our method is following. As pointed in [1, 4], a single truth table complementation will cause every $\mathbf{F}(w)$ to alter by ± 2. In terms of nonlinearity, it means, that the complementation any position will increase nonlinearity by $+1$ if $\mathbf{F}(w)$ altered by -2, and vice versa, will decrease nonlinearity by -1 if $\mathbf{F}(w)$ altered by $+2$. Thus, to modify a bent sequence to a highly nonlinear Boolean function we have to find the positions that if changed lead to a decrease in nonlinearity. Table 2 shows the quantity of positions we must change to design balanced Boolean functions along with the nonlinearity attained in that way. It is well seen that our simple direct modification allows constructing balanced Boolean functions with good nonlinearity. The values in the last row define the lowest bound of the nonlinearity attainable by direct modification (complementation) of the bent sequence. Nevertheless, the idea presented above allows only decrease in nonlinearity. The nonlinearity can be increased in a simple way. To increase nonlinearity we should not only find positions n^- complementation which leads to a decrease in nonlinearity, but then we should find positions n^+ complementation which leads to increase in nonlinearity. Naturally, the sum of these positions, n^- and n^+, is $2^{\frac{n}{2}-1}$. The following theorem allows calculate these values.

Theorem 2.1. *To construct a balanced Boolean function with maximal attainable nonlinearity* $\boldsymbol{nl}(f)_{max}$ *from a bent function, one needs to change* n^- *positions of the bent function that lead to decrease nonlinearity and* n^+ *positions of the bent function that lead to an increase in nonlinearity,*

$$n^- = \frac{n_{diff} + n_{need\ steps}}{2}, \text{ and} \tag{10}$$

$$n^+ = n_{need\ steps} - n^-. \tag{11}$$

where $n_{need\ steps} = 2^{\frac{n}{2}-1}$ *is a number of positions that we have to change to get a balanced sequence, and* n_{diff} *is a number of positions that we have to change to decrease the nonlinearity from* $\boldsymbol{nl}(f)$*, the upper bound of nonlinearity, to* $\boldsymbol{nl}(f)_{max}$*, the lowest upper bound (see Tables 1, 2).*

Proof. According to (6), (7), the difference between nonlinearity of the bent function and maximal attainable nonlinearity for the balanced function is 2. So, we have to change $n_{\text{diff}} = \mathbf{nl}(f_{\text{bent}}) - \mathbf{nl}(f)_{\max} = 2$ positions to obtain a decrease in nonlinearity from $\mathbf{nl}(f_{\text{bent}})$ to $\mathbf{nl}(f)_{\max}$. Changing another $n_{\text{need steps}} - n_{\text{diff}}$ positions must 'balance' each other: a complementation of the $\frac{(n_{\text{need steps}} - n_{\text{diff}})}{2}$ positions must produce a decrease in nonlinearity while the complementation of $\frac{(n_{\text{need steps}} - n_{\text{diff}})}{2}$ positions must produce increase in nonlinearity. Hance, n^- can be calculated as follows:

$$n^- = n_{\text{diff}} + \frac{n_{\text{need steps}} - n_{\text{diff}}}{2} = \frac{n_{\text{need steps}} + n_{\text{diff}}}{2}, \text{or} \tag{12}$$

$$n^- = 1 + \frac{n_{\text{need steps}}}{2}.$$

Accordingly, n^+ is calculated as $n^+ = n_{\text{need steps}} - n^-$. The Theorem 2.1 can be re-written in terms of Walsh-Hadamard transform. Here n_{diff} will be equal $\frac{|\mathbf{F}_{\text{bent}}(w) - \mathbf{F}_{\max}(w)|}{2}$. For example, Table 3 shows the quantity of positions we have to change to design balanced Boolean functions with the lowest upper bound, $\mathbf{nl}(f)_{\max}$ according to Theorem 2.1. □

It should be noted, that we failed to attain the lowest upper bound for $n = 8, 10, 12$ in that way.

TABLE 3. Attained nonlinearity $\mathbf{nl}(f)_{\max}$ by complementation

Method	Space dimension, n				
	4	6	8	10	12
Upper Bound of Nonlinearity, $\mathbf{nl}(f_{\text{bent}})$	6	28	120	496	2016
Lowest Upper Bound, $\mathbf{nl}(f)_{\max}$	4	26	118	494	2014
Needed positions $\frac{n^-}{n_+}$	2/0	3/1	5/3	9/7	17/15

2.1. Desired Nonlinearity

As we can see from Tables 1 and 2, there is a set of values lying between the lowest upper bound, $\mathbf{nl}(f)_{\max}$, and the nonlinearity attained by the direct complementation, $\mathbf{nl}(f)_{\min}$. Thus, for example, we can construct functions with $\mathbf{nl}(f) = 112, 114, 116$ for $n = 8$. Generalization of Theorem 2.1 allows us construct nonlinear balanced Boolean functions with the desired nonlinearity.

Theorem 2.2. *To construct a balanced Boolean function with desired nonlinearity $\boldsymbol{nl}(f_{des})$, $\boldsymbol{nl}(f)_{min} \leq \boldsymbol{nl}(f_{des}) \leq \boldsymbol{nl}(f)_{max}$, from a bent function, we need to change positions of the bent function that lead to a decrease in nonlinearity and positions of the bent function that lead to an increase in nonlinearity, where*

$$n^- = \frac{\boldsymbol{nl}(f_{bent}) - \boldsymbol{nl}(f_{des}) + n_{des}}{2}, \textit{ and} \tag{13}$$

$$n^+ = n_{need\ steps} - n^-. \tag{14}$$

Proof. To decrease of nonlinearity in the bent function from $\mathbf{nl}(f_{\text{bent}})$ to desired nonlinearity $\mathbf{nl}(f_{\text{des}})$, we have to change $n_{\text{diff}} = \mathbf{nl}(f_{\text{bent}}) - \mathbf{nl}(f_{\text{des}})$ positions that if changed affect the nonlinearity. Complementation of another $n_{\text{need steps}} - n_{\text{diff}}$ positions must 'balance' each other: complementation of $\frac{n_{\text{need step}} - n_{\text{diff}}}{2}$ positions that if changed affect the nonlinearity must decrease the nonlinearity and the complementation of $\frac{n_{\text{need step}} - n_{\text{diff}}}{2}$ positions that if changed produce an increase in nonlinearity must increase nonlinearity. Hence, n^- can be calculates as follows:

$$\begin{aligned} n^- &= n_{\text{diff}} + \frac{n_{\text{need steps}} - n_{\text{diff}}}{2} = \mathbf{nl}(f_{\text{bent}}) - \mathbf{nl}(f_{\text{des}}) + \\ &+ \frac{n_{\text{need steps}} - (\mathbf{nl}(f_{\text{bent}}) - \mathbf{nl}(f_{\text{des}}))}{2} = \\ &= \frac{\mathbf{nl}(f_{\text{bent}}) - \mathbf{nl}(f_{\text{des}}) + n_{\text{need steps}}}{2}. \end{aligned}$$

TABLE 4. Attained nonlinearity $\mathbf{nl}(f_{\text{des}})$ by complementation

Method	Space dimension, n				
	4	6	8	10	12
Upper Bound of Nonlinearity, $\mathbf{nl}(f_{\text{bent}})$	6	28	120	496	2016
Best known example	4	26	116	492	2010
Needed positions $\frac{n^-}{n_+}$	2/0	3/1	6/23	10/6	19/13

Accordingly, n^+ is calculated as $n^+ = n_{\text{need steps}-n^-}$. □

In the terms of the Walsh-Hadamard transformation the equation 12 can be re-writing as follows

$$n^- = \frac{\mathbf{F}_{\text{des}}(w) - \mathbf{F}_{\text{bent}}(w) + n_{\text{need steps}}}{4}. \tag{15}$$

As example Table 4 shows quantity of positions we have to change to design the balanced Boolean functions with desired nonlinearity $\mathbf{nl}(f_{\text{des}})$. Here $\mathbf{nl}(f_{\text{des}})$ is the best known example.

2.2. Desired Autocorrelation

The presented method can be adapted to design not only Boolean function with the desired nonlinearity, but with the desired autocorrelation also. Cost function $\mathbf{cost}(f)$ should be used like [5].

3. Results

To implement the described technique we generate a set of bent sequences. The pool sizes were $k = 100$, $k = 1000$, $k = 10000$. A sequence from the set was extracted and our technique was applied to that sequence. After the sequence was modified to meet the required parameters (balancedness, nonlinearity, and autocorrelation), the next sequence was extracted, and so on. The data presented below show the best achieved results comparing with the best known ones and exploitation characteristics of the method.

3.1. Strength results

In this subsection we detail the results of applying our modified method. Table 5 shows the best known profiles $(n, \deg, \mathbf{nl}, AC)$ constructed with heuristic techniques (note that there are another types of construction methods allow to attain the same results

Table 5. Best values $(n, \deg, \mathbf{nl}, AC)$ obtained using NTL, ACT and MHC

Method	Space dimension, n	
	6	8
NTL [6]	$(6, 5, 26, 16)$	$(8, 7, 116, 24)$ $(8, 5, 112, 16)$
ACT [6]	$(6, 5, 26, 16)$	$(8, 7, 116, 24)$ $(8, 5, 112, 16)$
Our MHC	$(6, 5, 26, 8)$	$(8, 7, 116, 24)$

Method	Space dimension, n	
	10	12
NTL [6]	$(10, 9, 486, 72)$ $(10, 9, 484, 64)$	$(12, 10, 1992, 156)$ $(12, 10, 1990, 144)$
ACT [6]	$(10, 9, 484, 56)$	$(12, 11, 1986, 128)$
Our MHC	$(10, 9, 488, 40)$	$(12, 11, 2002, 72)$

Table 6. Comparing the nonlinearity of balanced functions

Method	Space dimension, n				
	4	6	8	10	12
Lowest Upper Bound	4	26	118	494	2014
Best known example	4	26	116	492	2010
Dobbertin's Conjecture	4	26	116	492	2010
Bent Concatenation	4	24	112	480	1984
Random	-	-	112	472	1954
Random Plus Hill Climbing	-	-	114	476	1960
Genetic Algorithms	-	26	116	484	1976
NTL	-	26	116	486	1992
ACT	-	24	116	484	1986
Simulated Annealing [5]	4	26	116	484	1990
Our MHC	4	26	116	488	2002

[9,10]; here we discuss only heuristic techniques). Our method allows achieving the highest strength criteria. As one can see the lowest autocorrelation is attained for the all n. Additionally the attained nonlinearity is slightly higher for $n = 10$, $n = 12$. For $n = 10 \div 12$ the profiles are the best known for this time. All data from the Table 6 except the two latest rows are taken from [6]. All data from the Table 7 except the latest row are taken from [6].

TABLE 7. Conjectured bounds and attained values for autocorrelation of balanced functions

Method	Space dimension, n			
	6	8	10	12
Zhang and Zheng	16	24	48	96
Maitra Construction	16	24	40	80
Maitra Conjecture	16	24	40	80
NTL	16	16	64	144
ACT	16	16	56	128
Our MHC	8	24	40	72

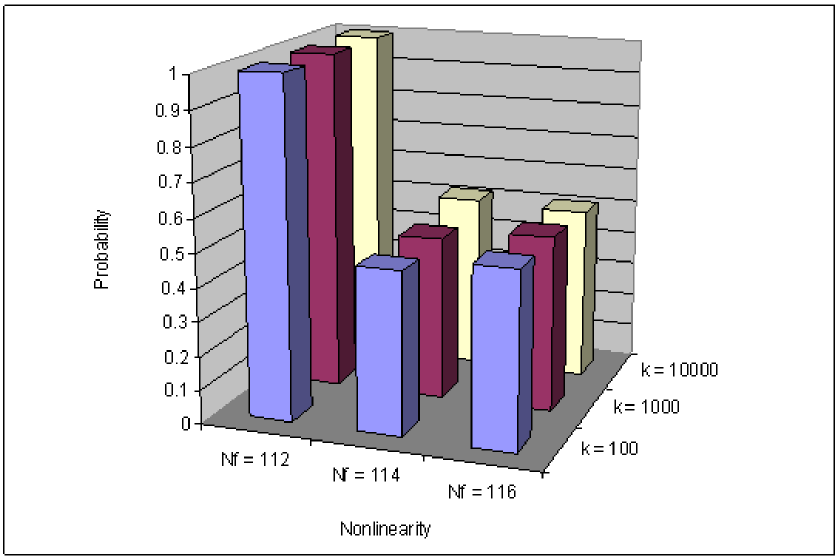

FIGURE 1. Probability of constructing Boolean functions with the required nonlinearity

3.2. Exploitation results

In this subsection we detail the exploitation results of applying of our modified method. Here, we presented the results only for $n = 8$. Note, that it is possible situation when the function with the desired criterion cannot be constructed with the first time. It may happen in the case that the modified sequence does not have Improve Set [1,4]. Figures 1, 2 show the probability of constructing the Boolean functions with the required nonlinearity and the required autocorrelation for each of the set correspondingly. It

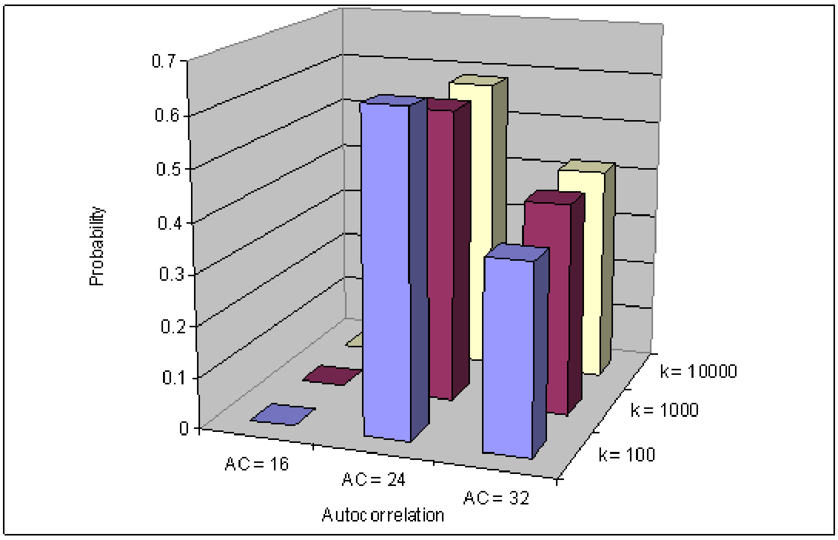

Figure 2. Probability of constructing Boolean functions with the required autocorrelation

can be seen that constructing a Boolean function with nonlinearity $\mathbf{nl}(f)_{\min}$, $\mathbf{nl}(f) = 12$ for this case, is too easy and occurs with probability equal 1. Generating Boolean functions with nonlinearity $\mathbf{nl}(f) = 114$, $\mathbf{nl}(f) = 116$ leads to success with probability closed to 0.5 for both cases. The generated functions have low autocorrelation $AC_f = 24 \div 32$.

To guarantee constructing the Boolean functions with the required nonlinearity, the iteration procedure was applied. The Table 8 shows quantity of iterations we required to construction Boolean functions with the required nonlinearity. Analysis of Table 8 shows that more than 90% of all the constructed functions with required nonlinearity can be obtained with only but four iterations. In Figure 3 we show the probability representation of the construction Boolean function with $\mathbf{nl}(f) = 116$ for the all pool sizes.

TABLE 8. Conjectured bounds and attained values for autocorrelation of balanced functions

I	Nonlinearity								
	112			114			116		
	100	1000	10000	100	1000	10000	100	1000	10000
1	97	978	9699	86	879	8741	52	518	5093
2	0	2	208	12	105	1084	25	243	2487
3	2	15	40	2	12	158	14	133	1256
4	1	1	21		2	15	4	60	601
5		2	19			2	4	22	299
6		2	9				0	14	129
7			4				1	7	69
8								1	35
9								2	18
10									6
11									6
12									1

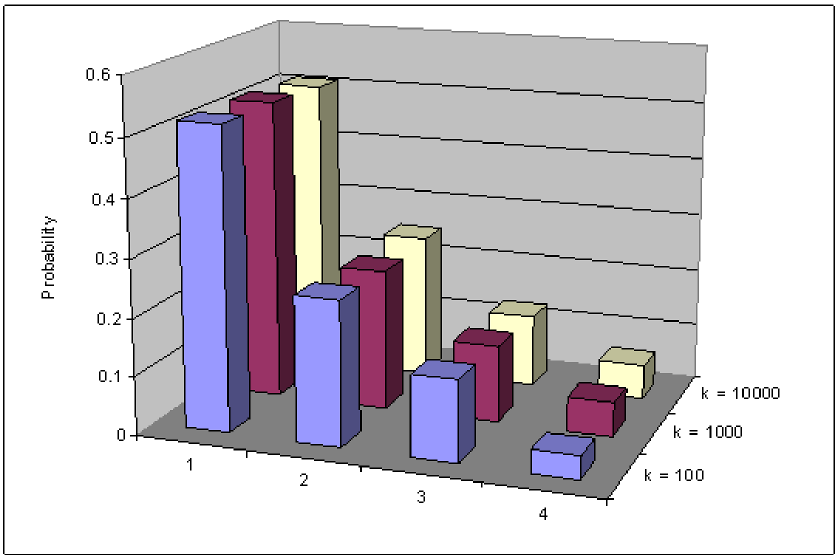

FIGURE 3. Probability of construction Boolean function with $\mathbf{nl}(f) = 116$

4. Conclusion

A new heuristic approach to design of the cryptographically strong Boolean functions is considered. The shown technique allows designing the functions that possess good cryptographic properties. The functions that have the best known profiles have been designed. The Appendix A contains the obtained Boolean functions (truth tables) with the best known profiles in binary format.

References

[1] W. Millan, A. Clark, E. Dawson. *Smart Hill Climbing Finds Better Boolean Functions*, In Proceedings of the *Workshop on Selected Areas on Cryptography, SAC'97*. Springer-Verlag (1997), 50-63

[2] W. Millan, A. Clark, E. Dawson. *An Effective Genetic Algorithm for Finding Highly Non-linear Boolean Functions*. In Proceedings of the *First International Conference on Information and Communications Securit* LNCS Vol. 1334. Springer-Verlag, Berlin Heidelberg New York (1997), 149-158.

[3] W. Millan, A. Clark, E. Dawson. *Heuristic Design of Cryptographically Strong Balanced Boolean Functions*. In Proceedings of *Advances in Cryptology - EUROCRYPT'98*. LNCS Vol. 1403. Springer Berlin Heidelberg New York (1998), 489.

[4] W. Millan, A. Clark, E. Dawson. *Boolean Function Design Using Hill-climbing Methods*. In Proceedings of the *4th Australian Conference on Security and Information Privacy, ASCIP'99*. Wollongong, NSW, Australia, April 1999. LNCS Vol. 1587. Springer Berlin Heidelberg (1999), 1-11.

[5] J.A. Clark, J.L. Jacob. *Two-Stage Optimisation in the Design of Boolean Functions*. In Proceedings of the *5th Australasian Conference, ACISP'00*. Brisbane, Australia, July 10-12. LNCS Vol. 1841. Springer-Verlag (2000), 242-254.

[6] J. Clark, J.L. Jacob, S. Stepney, S. Maitra and W. Millan. *Evolving Boolean Functions Satisfying Multiple Criteria*. In Proceedings of the *Advances in Cryptology - INDOCRYPT'02*. LNCS Vol. 2551, Springer (2002), 246-259.

[7] W. Maier, O. Staffelbach. *Nonlinearity criteria for cryptographic functions*. In Proceedings of the *Advances in Cryptology - EUROCRYPT'89: Workshop on the Theory and Application of Cryptographic Techniques*. LNCS Vol. 2551, Springer (2002), 246-259.

[8] J. Seberry, X.-M. Zhang and Y.Zheng. *Nonlinearity and Propagation Characteristics of Balanced Boolean Functions. Information and Computation* LNCS Vol. 2551, Springer (2002), 246-259.

[9] P. Sarkar, S. Maitra. *Construction of nonlinear Boolean functions with important cryptographic properties*. In Proceedings of the *Advances in Cryptology-EUROCRYPT'00*. LNCS vol. 1807, Berlin, Germany: Springer-Verlag, (2000), 491-512.

[10] S. Maitra, E. Pasalic. *Further constructions of resilient Boolean functions with very high nonlinearity. IEEE Trans. Inform. Theory.* Vol. 48, (2000), 1825-1834.

The authors would like to thank the Oksana Lyutikova for her helpful translation, which improved significantly the presentation of the paper.

5. **Appendix**

$(8, 7, 116, 24)$ profile:

0110011011100110011001101001100101101011100101100000000
0000000001010101010111110110100101101001000011110000 11
1100001111111100000011110000111100001100111100110101110
0110011011100111100110000110000000011111111010111101010
0101010101011010101001011010010110 10

$(10, 9, 488, 40)$ profile:

0110101001101010011010100111101101101010100101010110101
0100001000011111100111111001111110010111000111111110000
0000111111110100010110101001101000100101011000010001101
0101001010110010101011110110011011100111111110000001101
0001001111111100000011000000001011100110101001101010011
0100001111011011010101001010001101010100001000011111100
1111110011111100101110001111111100000000111111110100010
1101010011010101001010110000100011010101001010110010101
0111101100111111001101111110000001101000100111111100000
0110000000010111001101010011010100110101001110110 11010
1010010101011010101000000000111111001111110011111100101
1100011111111000000000111111110100010110101001101010 1001
0101100001000110101000010101100101010101101000110111001
1111111000000110100010011111111000000110000000010111010
0101011001010110010101100001001001010101101010100101010
1111011110000001100000011000000110100011100000000111111
1100000000000101000010101100101010110101001111011 1001010
1011010100110100010000100110000001100000000111111001001
1011000000000111011001111111111010001

$(12, 11, 2002, 72)$ profile:

0000000011001100000000000011001100101010100110011010 10101
0011001101111111100110011111111110011001101010101100110
0101010101100110011111111110011001111111111001100110 1010
1011001100101010101100110011111111110011001111111111 0011
0011010101011001100101010101100110010000000011001100000

0000011001100101010100110011010101010011001101111111100
1100111111111100100011010101011001100101010101100110011
1111101001100111111111001100110101010110011001010101010101
100110011111101100110011111111110011001101010101011001100
1010101011001100111110000001111001111000000111100010110
101001011001011010100101100000111111000011000011111000
0111010010101101001101001010110100100001011110000110000
1111110000101010010101101001101001010110100100001111110
0001100001111110000111010010101101001101001010110100111
1100000011110011100000001111000101101010010110010110101
0010110000011111100001100001111110000111010010101101001
1010010101101001000011111100001100001111110000111010000
1011010011010010101101001000011111100001100001111110000
111010010101101001101001010110000111111111001100111111
111001100110101010110011001010101011001100100000000001100
11000000000011001100101010100110011010101010011001100110000
00000110011000000000001100110010101010011001101010101001
10011000000000001001100000000000110011001010101001100110101
01010100100011000000000110011000000000001100110010101010
0110011010101010011001101111111110011001111111111110011001
10101010110011001010101011001100111111111100110001111111
11001100110101010110011001010101011001100111111110100110
0111111111100110011010101011001000101010101100110010000
1111110000110000111111000011101001010110100110100101011
0100111110000001111001111000000111100010110101001011001
0110101001011011110000001111001111000000111100010110101
0010110010110101001011011110000001111001111000000111100
0101101010010110010110101001011011110000001111001111000
0001111000101101010010110010110101001011000001111110000
110000111111000011101001010110100110100101011010010000 1
1111100001100001111110000111010010101101001101001010110
1001000011111100001100001111110000111010010101101001101
001010110100100000000011001100111111110011001100101010 01
10011001010101100110011111111110011001100000000110011000
10101011001100110101010011001101111111110011001100000000 0
11001100010101011001100110101010011001101111111110011001
100000000110011000101010110011001101010100110011000000 00
00110011001111111110011001110101010011001100101010110001
001111111110011001100000000110011000101010110011001101 0
1010011001101111111110011001100000000110011000101010110 0
1100110101010011001001011111110011001100000000110011000 1

0101011001000110101010011001101111000000111100000011111
1000010010110101001011010100101011010010000111111000011
1111000000111100101001010110100101011010100101100000111
1110000111111000000111100101001010110100101011010100101
1000001111110000101111000000111100101001010110100101011
0101001011011110000001111000000111111000011010110101001
0110101001010110100100001111110000111111000000111100101
0010101101001010110101000011000001111110000111111000000
1111000010010110010110010110100110100100001111110000110
1110000001111001010010110010110010110100110100111111111
0011001100000000110011000101010110011001101010100010011
0000000001100110011111111001100111000101001100110010101
0110011001000000001100110011111111001100111010101001100
1100101010110011001000000001100110011111111001100111010
1010011001100101010110011001000000001100110011111111000
1001110101010011001100101010110011001111110110011001100
0000001100110000010101100110011010101001100110111111110
0110011000000001100110001010101100110011010101001100110
1111111100110011000000001100110001010101100110011010101
0011001100000111111000011111100000011110010100101011010
0101011010100101101101000000111100000011111100001101011
0101001011010100101011010011110000000111100000011111100
0011010110101001011010100101011010011111000000111100000
0111011000011010110101001011010100101011010011111000000
1111000000111111000011010110101001011010100101011010010
0001111110000111111000000111100101001010110100101011010
1001011000001111110000111111000000111100101001011001011
0010110100110100100001111110000111111000000111100101001
001001011001011010011010011