Cryptographically Strong S-boxes Generated by Modified Immune Algorithm *

Georgi Ivanov[1], Nikolay Nikolov[2], and Svetla Nikova[3]

[1] Institute of Mathematics and Informatics, Bulgarian Academy of Sciences
[2] Institute of Mathematics and Informatics, Bulgarian Academy of Sciences
[3] KU Leuven, Dept. ESAT/COSIC and iMinds, Belgium

Abstract. S-boxes play an important role in ensuring the resistance of block ciphers against cryptanalysis as often they are their only nonlinear components. The cryptographic properties of S-boxes and a variety of constructions have been studied extensively over the past years. Techniques for S-box generation include algebraic constructions, pseudo-random generation and heuristic approaches. The family of artificial immune algorithms is a particular example of a heuristic approach. In this paper we propose an S-box generation technique using a special kind of artificial immune algorithm, namely the clonal selection algorithm, combined with a slightly modified hill climbing method for S-boxes. Using this special algorithm we generate large sets of highly nonlinear bijective S-boxes of low differential uniformity in a reasonable search time.

Keywords: immune algorithms, S-boxes, nonlinearity, differential uniformity

1 Introduction and Motivation

Most block ciphers in modern cryptography contain one or more non-linear components that have the ability to provide the effect of confusion [26], which is of vital importance for the strength of the cipher. Often these components are n to m Boolean mappings, the so called S-boxes. Among the whole set of S-boxes the bijective ones are particularly interesting.

In order to improve the resistance of a block cipher to *linear* and *differential cryptanalysis* [2,3,17], two main approaches are used - either by increasing the number of active S-boxes or by using S-boxes with strong cryptographic properties. The most important cryptographic properties an S-box should possess are *regularity*, high *nonlinearity*, high *algebraic degree*, low *differential uniformity* and low *autocorrelation*. The known techniques for the construction of S-boxes can be divided into three main streams: *algebraic constructions*, *pseudo-random generation* and *heuristic techniques*.

The first approach is the most popular one, where the S-boxes are designed according to some proven mathematical relations and principles. The most famous representatives of this approach are the bijective $(n \times n)$ S-boxes based on inversion in the finite field $GF(2^n)$. They are the best S-boxes found and are simultaneously optimal with respect to most of the desired criteria. The best values achieved for nonlinearity and differential uniformity using algebraic constructions are $N_{inv} = 2^{n-1} - 2^{\frac{n}{2}}$ and $\delta_{inv} = 4$, [21]. The (8×8) S-box of AES [10] is such an S-box, which has high algebraic degree - 7, high nonlinearity - 112, low autocorrelation - 32 and low differential uniformity - 4. Although these S-boxes are often preferred because of their excellent cryptographic properties, there are some concerns related to their simple

* The research is done as a part of the project "Finite geometries, coding theory and cryptography" between the Research Foundation - Flanders (FWO) and the Bulgarian Academy of Sciences

algebraic structure and possible future vulnerability to algebraic attacks [9]. Furthermore, the number of these S-boxes is small and all of them are affine equivalent.

The *pseudo-random* S-box generation technique consists of constructing the S-box from a table of random numbers followed by its conformity testing. This approach is doomed to failure from the very beginning as most of the desired cryptographic criteria are often contradictory to each other, which greatly reduces the number of S-boxes that are good with respect to all criteria and diminishes the probability of picking up a good S-box.

Heuristic algorithms are used for S-box generation in a process of iteratively improving an S-box or a whole set of S-boxes with respect to one or more properties. In contrast to the algebraic constructions, heuristic techniques are able to produce big sets of S-boxes as they use direct search methods. Most commonly the cryptographic properties of S-boxes obtained by heuristic algorithms are not as good as the ones of the algebraically constructed S-boxes. However, in recent years the difference between these properties is getting more and more indistinguishable. The latter is achieved by using some specific heuristic techniques like the hill climbing method, the simulated annealing method, the genetic algorithms or different combinations of these. With respect to nonlinearity, the highest value achieved in the case of (8×8) bijective S-boxes by: the hill climbing method is 100 [18]; the *simulated annealing method* is 102 [8]; and by a special genetic algorithm is 104 [27]. Recently, in [16], values of 104 for nonlinearity were achieved by a method, referred to as the *modified gradient descent*, based on swapping certain number of values in a permutation. In [15], values of 112, equal to the ones of the S-box of AES, for nonlinearity were achieved by a method, referred to as the *reversed genetic algorithm*, starting from initial population full of S-boxes based on inversion in the finite field and obtaining large sets of *non-equivalent* S-boxes. As regards to differential uniformity, the lowest value achieved in the case of (8×8) *bijective* S-boxes is 4, [15, 22, 23]. Although most of the methods described give good results for constructing bijective S-boxes with respect to only one of the main criteria, it becomes much more challenging when both nonlinearity and differential uniformity should be considered simultaneously. In [8, 18, 27] differential uniformity is not considered at all. In [16] the S-box reported has nonlinearity 104 and differential uniformity 8. In [22, 23] the best value obtained for differential uniformity is 4, while the respective nonlinearity value is 98. In [15] combinations $(112, 6)$ and $(110, 4)$ for nonlinearity and differential uniformity are reported.

1.1 Contribution

In this paper we propose a new *heuristic method* for generation of big sets of $(n \times n)$ *bijective* S-boxes possessing a good combination of the target properties like: high nonlinearity, high algebraic degree, low differential uniformity and low autocorrelation. The method, referred to as the *SpImmAlg*, is based on using a specific *artificial immune algorithm* (section 7.2 of [4]) combined with a modification of the *hill climbing method* for S-boxes [18]. The input of *SpImmAlg* is an $(n \times n)$ bijective S-box, which can either be pseudo-randomly generated or some other S-box possessing specific properties. In the case of random input and $n = 8$, the output of the algorithm after 10 days work on a cluster with 32 cores is a large set of thousands of (8×8) bijective S-boxes with nonlinearity 104 and differential uniformity 6. Compared to the finite field inversion-based S-boxes and the ones obtained in [13, 15], these

S-boxes seem to be worse, but after considering the fact that the former either are affine equivalent to the finite field inversion-based S-boxes or at least share to a certain extent their algebraic structure, then the latter S-boxes appear to be much more attractive due to their random origin and the expected better resistance to algebraic attacks [9].

2 Preliminaries

In this section we recall some basic definitions and properties of Boolean functions. We refer to [5,6] for a comprehensive survey on Boolean functions.

Let $S : \mathbb{B}^n \to \mathbb{B}^m$ be an n-binary input m-binary output mapping, referred to as an $(n \times m)$ S-box. Then, to each $x = (x_1, x_2, \ldots, x_n) \in \mathbb{B}^n$ some $y = (y_1, y_2, \ldots, y_m) \in \mathbb{B}^m$ is assigned by $S(x) = y$, where $\mathbb{B} = \{0, 1\}$ is the 1-dimensional Boolean space. The $(n \times m)$ S-box S can be considered as a vectorial Boolean function comprising m individual Boolean functions $f_1, f_2, \ldots, f_m : \mathbb{B}^n \to \mathbb{B}$, where $f_i(x) = y_i$ for $i = 1, 2, \ldots, m$. These functions are referred to as the *coordinate* Boolean functions of the S-box. It is well known that most of desirable cryptographic properties of the S-box can be defined also in terms of all non-trivial linear combinations of the coordinate functions, referred to as the S-box *component* Boolean functions: $g_c : \mathbb{B}^n \to \mathbb{B}$, where $g_c = c_1 f_1 \oplus c_2 f_2 \oplus \ldots \oplus c_m f_m$ and $c = (c_1, c_2, \ldots, c_m) \in \mathbb{B}^m \setminus \{0\}$.

2.1 Boolean functions

A Boolean function $f : \mathbb{B}^n \to \mathbb{B}$ can be represented by its binary output vector containing 2^n elements, referred to as the *truth table*. The function can also be represented by its polarity truth table, where instead of f the signed function $\widehat{f} = (-1)^f$ is considered.

Another way of representing f is by its *algebraic normal form*, denoted by ANF_f:

$$ANF_f = a_0 \oplus a_1 x_1 \oplus a_2 x_2 \oplus \ldots \oplus a_n x_n \oplus a_{1,2} x_1 x_2 \oplus \ldots \oplus a_{1,2,\ldots,n} x_1 x_2 \ldots x_n,$$

where $a_I \in \mathbb{B}, I \subseteq \{1, 2, \ldots, n\}$. The *algebraic degree* of f, denoted by $deg(f)$, is the number of variables of the largest product term of ANF_f having a non-zero coefficient.

The *Walsh-Hadamard transform (WHT)* of an n-variable Boolean function f, represented by its polarity form $\widehat{f}$, is denoted by $\widehat{F}_f(w)$ and defined as:

$$\widehat{F}_f(w) = \sum_{x \in \mathbb{B}^n} \widehat{f}(x)(-1)^{<w,x>} = \sum_{x \in \mathbb{B}^n} (-1)^{f(x) \oplus <w,x>} = \sum_{x \in \mathbb{B}^n} \widehat{f}(x)\widehat{l}_w(x),$$

where $\widehat{l}_w(x)$ is the signed function of the linear function $l_w(x) =< w, x >$.

$\widehat{F}_f(w) \in [-2^n, 2^n]$, $\forall w \in \mathbb{B}^n$, and $\widehat{F}_f(w)$ is known as a *spectral Walsh coefficient*, while the real-valued vector of all 2^n spectral coefficients is referred to as the WHT Spectrum. The maximum absolute value, taken by $\widehat{F}_f$, is given by: $WHT_{max}(f) = \max_{(w \in \mathbb{B}^n)} |\widehat{F}_f(w)|$.

The *autocorrelation transform (ACT)*, taken with respect to $\alpha \in \mathbb{B}^n$, of an n-variable Boolean function f, represented by its polarity form $\widehat{f}$, is denoted by $\widehat{r}_f(\alpha)$ and defined as:

$$\widehat{r}_f(\alpha) = \sum_{x \in \mathbb{B}^n} (-1)^{f(x) \oplus f(x \oplus \alpha)} = \sum_{x \in \mathbb{B}^n} \widehat{f}(x)\widehat{f}(x \oplus \alpha)$$

Thus, $\widehat{r}_f(\alpha) \in [-2^n, 2^n]$, $\forall \alpha \in \mathbb{B}^n$, and $\widehat{r}_f(0) = 2^n$. $\widehat{r}_f(\alpha)$ is known as a spectral autocorrelation coefficient of the function, while the real-valued vector of all 2^n autocorrelation coefficients is referred to as its *ACT Spectrum*.

Three of the important cryptographic properties that any n-variable Boolean function f should possess are *balance*, high *nonlinearity* and low *autocorrelation*.

f is balanced if $w_H(f) = 2^{n-1}$, where $w_H(f)$ denotes the Hamming weight of f.

The nonlinearity of f, denoted by N_f, is the Hamming distance between f and the set $A(n)$ containing all n-variable affine Boolean functions. It is given by:

$$N_f = 2^{n-1} - \frac{1}{2} WHT_{max}(f) = 2^{n-1} - \frac{1}{2} max_{(w \in \mathbb{B}^n)} |\widehat{F}_f(w)|$$

The *absolute indicator* of f, denoted by $AC(f)_{max}$, is defined as the maximal non-trivial absolute autocorrelation value. It is given by: $AC(f)_{max} = max_{(\alpha \in \mathbb{B}^n \setminus \{0\})} \ |\widehat{r}_f(\alpha)|$.

Two n-variable Boolean functions f and g are *affine equivalent* if and only if there exist some invertible $(n \times n)$ binary matrix A, vectors $b, c \in \mathbb{B}^n$ and a scalar $d \in \mathbb{B}$, such that $g(x) = f(Ax \oplus b) \oplus < c, x > \oplus d$.

2.2 Vectorial Boolean functions

All Boolean function properties discussed so far can be extended to the case of vectorial Boolean functions (S-boxes). The important difference in the manner the S-box properties are derived consists in that it is necessary the properties of the S-box component Boolean functions to be considered rather than the coordinate Boolean functions only.

Any good S-box should be *regular* (balanced) in order to avoid trivial statistical attacks. An $(n \times m)$ S-box S with $n \geq m$ is said to be regular if, for each its output $y \in \mathbb{B}^m$, there are exactly 2^{n-m} inputs that are mapped to y. Clearly, each *bijective* $(n \times n)$ S-box S is always regular since it represents a *permutation*. It is well known that an $(n \times m)$ S-box with $n \geq m$ is regular if and only if all its component Boolean functions are balanced [25].

In order to improve the cipher immunity against linear cryptanalysis, [17], any $(n \times m)$ S-box S should have small magnitude entries in its *linear approximation table* LAT_S, which is shown to be equivalent in [24] to the statement that the *nonlinearity* of each component Boolean function of S should be high. Thus, the nonlinearity of S, denoted by N_S, is given by the minimal nonlinearity among the nonlinearities of the component Boolean functions:

$$N_S = min_{(c \in \mathbb{B}^m \setminus \{0\})} \ g_c = min_{(c=(c_1,c_2,\ldots,c_m) \in \mathbb{B}^m \setminus \{0\})} \ N_{c_1 f_1 \oplus c_2 f_2 \oplus \ldots \oplus c_m f_m}$$

In order to resist *low order approximation* attacks each $(n \times m)$ S-box S should have an algebraic degree as high as possible, [14, 20]. The *(minimal) algebraic degree* of S, denoted by $deg(S)$, is defined as the minimal algebraic degree among the degrees of the component Boolean functions. It can be expressed as follows:

$$deg(S) = min_{(c \in \mathbb{B}^m \setminus \{0\})} \ deg(g_c) = min_{(c=(c_1,c_2,\ldots,c_m) \in \mathbb{B}^m \setminus \{0\})} \ deg(c_1 f_1 \oplus c_2 f_2 \oplus \ldots \oplus c_m f_m)$$

In order to improve the cipher immunity against *differential cryptanalysis*, [2], any $(n \times m)$ S-box S with $n \geq m$ should have small entries in its *difference distribution table* DDT_S, not counting the first entry in the first row [1, 7, 11]. That is, S should have as low *differential uniformity* as possible, where differential or δ-uniformity, denoted by δ, is defined by:

$$\delta = max_{(\alpha \in \mathbb{B}^n \setminus \{0\})} \; max_{(\beta \in \mathbb{B}^m)} \; |\{x \in \mathbb{B}^n | S(x) \oplus S(x \oplus \alpha) = \beta\}|$$

It is well known that δ takes always only even values in $[2^{n-m}, 2^n]$. Then, the smallest possible value of δ in the case of bijective S-boxes $(n = m)$ is 2. Such S-boxes are referred to as the *almost perfect nonlinear* (APN) permutations.

In order to improve the *avalanche effect* of the cipher, [12], any $(n \times m)$ S-box S should have low autocorrelation, that is, the *absolute indicators* of the component Boolean functions of the S-box should be as small as possible. In other words, the maximal absolute indicator among all the absolute indicators, denoted by $AC(S)_{max}$, should be as small as possible.

Thus, we are looking in this paper for $(n \times n)$ bijective S-boxes possessing high nonlinearity and high algebraic degree as well as low differential uniformity and low autocorrelation.

Both $(n \times n)$ *bijective* S-boxes S_1 and S_2 are said to be affine equivalent if there exist invertible affine permutations $A(x)$ and $B(x)$, such that $S_1 = B \circ S_2 \circ A$.

3 Heuristic techniques

There are different types of heuristic methods suitable for solving computational problems. Some of them are inspired by a physical process (simulated annealing), some - by the process of biological evolution (genetic algorithms), while other methods are inspired by the process and mechanisms of the biological immune system (immune algorithms). Related to S-box generation, all types of heuristic techniques share the same approach. It involves a process, where given S-box or a set of S-boxes are iteratively improved with respect to one or more of their properties until either some reasonable number of iterations is reached or some chosen in advance specific threshold values for these properties are achieved.

Hill climbing method consists in applying some small modifications of a certain number of distinct elements of a function in order one or more its cryptographic properties iteratively to be improved. Simulated annealing method involves a sort of extension to the hill climbing technique, allowing the searching process to move out of a local optimum in order to continue. Genetic algorithms work with populations of candidate solutions. Being iteratively subject to the three main operations inspired by the natural evolution - crossover, mutation and selection, these populations transform into better ones, possessing all the properties desired. Each new generation is formed by those individuals from the previous one that have passed the fitness test. The fitness test is based on using a fitness or a cost function, responsible for taking the decision whether the evaluated individuals will survive to the next generation or not. Only the fittest individuals (those of maximal fitness or of low cost respectively) survive. In contrast to genetic algorithms, immune algorithms work only with one candidate solution, corresponding to the most appropriate type of general immune cells (lymphocytes) that will fight a specific pathogen. Once the initial solution is selected, it starts proliferating. During

the proliferation process the initial solution is subject to some small copying errors, referred to as the somatic hypermutation. The main goal of the combined process is by this duplication and variation the adaptation and the fighting power of the initial solution to be improved or in other words, the genetic variation preventing from falling into a local optimum to be present. Then, the process is iterated by replacing the initial solution with a new one, which is selected as the fittest solution from the new generation by a fitness or a cost function like in the case of genetic algorithms. At the end, the fittest solution (the one of maximal fitness or respectively of the lowest cost) of the final generation is the best solution.

4 New method

The proposed new method is based on a heuristic algorithm, denoted by *SpImmAlg*, which combines a modified algorithm from the *artificial immune algorithms* family, referred to as *clonal selection algorithm* (section 7.2 of [4]), with a modification of the known *hill climbing method* for S-boxes [18]. The basic idea is to start from an S-box and iteratively to improve its cryptographic properties. The main goal of the algorithm is rapidly to be constructed a variety of strong $(n \times n)$ *bijective* S-boxes, which possess target cryptographic properties such as: high *nonlinearity*, high algebraic degree, low differential uniformity and low autocorrelation.

4.1 Algorithm description

SpImmAlg has as input an $(n \times n)$ bijective S-box S_0. The initial S-box S_0 can be of any type - either a pseudo-randomly generated S-box or a specific one possessing certain cryptographic properties. The modified hill climbing method is applied to S_0. It consists of repeatedly swapping any two elements of S_0 and calculating the *cost* of the newly obtained S-box S by a *cost* function $cost(\cdot)$. If $cost(S) < cost(S_0)$, the process iteratively continues form S on. Otherwise, a new S-box S is obtained and so on. The output S of the modified hill climbing method, possessing the lowest cost, is then forwarded as an input to the *modified clonal selection algorithm*. By means of two different *somatic mutation* functions - $mutation_1(\cdot)$ and $mutation_2(\cdot)$ it produces 4 new S-boxes S_1, S_2, S_3 and S_4. We use 4 S-boxes in order to speed up the execution, however any other number of "children" is also possible. Each of these S-boxes is then transformed into a better new one by the modified hill climbing method. The best S-box of the four with respect to the respective cost function is then forwarded to the input of the modified clonal selection algorithm and the process starts all over. The algorithm makes use of 5 main functions, all having as an input argument any $(n \times n)$ *bijective* S-box S. These are the three distinct cost functions - $cost_1(\cdot)$, $cost_2(\cdot)$ and $cost_3(\cdot)$, forming as a product the final *cost* function $cost(\cdot)$ that has to be minimized, and both *mutation* functions - $mutation_1(\cdot)$ and $mutation_2(\cdot)$ used in order to guarantee the wide variety of the new generations:

1. The first *cost* function $cost_1(\cdot)$ calculates the cost of S by the rule:

$$cost_1(S) = \sum_{c<d \in \mathbb{B}^n \setminus \{0\}} \sum_{\omega \in \mathbb{B}^n} \mid |\widehat{F}_{c_1 f_1 \oplus c_2 f_2 \oplus \ldots \oplus c_n f_n}(\omega)|^3 - |\widehat{F}_{d_1 f_1 \oplus d_2 f_2 \oplus \ldots \oplus d_n f_n}(\omega)|^3 \mid^7.$$

   The goal of this cost function is to increase the S-box nonlinearity by making the absolute WHT spectrum as flat as possible.

2. The second *cost* function $cost_2(\cdot)$ calculates the cost of S by the rule:

$$cost_2(S) = \sum_{c=(c_1,c_2,\ldots,c_n)\in\mathbb{B}^n\setminus\{0\}} \sum_{\omega\in\mathbb{B}^n} |\widehat{F}_{c_1f_1\oplus c_2f_2\oplus\ldots\oplus c_nf_n}(\omega) - 21|^7.$$

The cost function $\sum_{c=(c_1,c_2,\ldots,c_n)\in\mathbb{B}^n\setminus\{0\}} \sum_{\omega\in\mathbb{B}^n} |\widehat{F}_{c_1f_1\oplus c_2f_2\oplus\ldots\oplus c_nf_n}(\omega) - X|^R$ for X, R real was first proposed in [8]. Later in [27] many pairs of parameters have been investigated for (8×8) S-boxes and the pair $X = 21, R = 7$ was shown to achieve the best results.

3. The third *cost* function $cost_3(\cdot)$ calculates the cost of S by the rule:

$$cost_3(S) = \sum_{\delta_{11}\neq\delta_{ij}\in DDT_S} (\delta_{ij}-1)^2.(\delta_{ij}-2)^2.(\delta_{ij}-4)^2, \text{ where}$$

δ_{ij} $(i, j = 1, 2, \ldots, 2^n)$ are the elements of the difference distribution table DDT_S of S. Naturally this cost function achieves its minimum when the δ-uniformity is 2 or 4.

4. The final cost function $cost(\cdot)$ calculates the overall cost of S by the rule:

$$cost(S) = cost_1(S).cost_2(S).cost_3(S),$$

i.e. by combining (and minimizing) the 3 cost functions.

5. The first mutation function $mutation_1(\cdot)$ transforms S into S' by the rule:

$$S' = mutation_1(S), \text{ where}$$

S' is obtained from S by swapping two its neighboring elements. The positions p and $p-1$ of the elements that are going to be swapped depend on the number p in the range from 2 to 2^n, which is chosen at random at each execution of the function. The latter serves as a guarantee that whenever the mutation function $mutation_1(\cdot)$ is being executed, the resulting S-box will be different from the previous ones.

6. The second *mutation* function $mutation_2(\cdot)$ transforms S into S'' by the rule:

$$S'' = mutation_2(S), \text{ where}$$

S'' is obtained from S in the following manner: a block of an arbitrary length q in the range from 2 to 8 of neighboring elements is modified in S. If q is an even number, then the elements that are symmetric with respect to the position p, which splits the block into two parts of equal length, are swapped. Otherwise, the element in the middle of the block stays in place, while all the other elements that are symmetric with respect to this element are swapped. In both cases, due to the random choice of q and p, the resulting S-box again will differ all the previous ones at each execution of the function. It should be noted that $mutation_2$ is an extension of $mutation_1$ (case $q = 2$). In general different values of q can be used, our experiments for (8×8) S-boxes show that the best results can be expected when q is upper bounded by 8.

4.2 Algorithm pseudo-code

STEP 1 (Initialization)

– Define an integer n, representing the dimensions of desired $(n \times n)$ *bijective* S-boxes.
– Generate a random $(n \times n)$ *bijective* S-box S_0.

STEP 2 (Initial selection)
– Start the modified hill climbing method (MHCM) with S_0 as an input.
– In result, obtain the $(n \times n)$ bijective S-box S, which has the lowest cost: $cost(S) = cost_1(S).cost_2(S).cost_3(S)$, $S = MHCM(S_0)$.

STEP 3 (Somatic hypermutation)
– Twice apply the *mutation* function $mutation_1(\cdot)$ with S as an input: $S_1 = mutation_1(S)$ and $S_2 = mutation_1(S)$.
– Twice apply the *mutation* function $mutation_2(\cdot)$ with S as an input: $S_3 = mutation_2(S)$ and $S_4 = mutation_2(S)$.
– Obtain 4 different $(n \times n)$ bijective S-boxes S_1, S_2, S_3 and S_4.

STEP 4 (Selection)
– Start the modified hill climbing method for each of S_1, S_2, S_3 or S_4 as an input.
– In result, obtain four low-cost $(n \times n)$ bijective S-boxes S'_1, S'_2, S'_3 and S'_4 : $S'_1 = MHCM(S_1)$, $S'_2 = MHCM(S_2)$, $S'_3 = MHCM(S_3)$ and $S'_4 = MHCM(S_4)$.
– Compare the overall cost, $cost(\cdot)$, of each of the four S-boxes S'_1, S'_2, S'_3, S'_4 and set S' to be the one with the lowest cost.

STEP 5 (Stopping criterion)
– If some chosen in advance threshold number of iterations or execution time e.g. 10 days is reached, then STOP.
– Otherwise, set S to S' and go to step 3.

5 Experimental results

In this section the results obtained by *SpImmAlg* are provided. The goal was to produce a variety of (8×8) *bijective* S-boxes with main cryptographic properties as close as possible to the finite field inversion-based S-boxes starting from a random S-box. For the sake of simplicity, we chose as a criterion to stop the algorithm after it runs for 10-days (instead of some threshold number of iterations).

5.1 Results obtained by *SpImmAlg* in the case of (8×8) *bijective* S-boxes

We ran *SpImmAlg* with a randomly generated (8×8) bijective S-box S_0. As a result, in 10 days neither APN permutations nor S-boxes better than the finite field inversion-based ones were found. The majority (35 000) of the obtained (8×8) bijective S-boxes have nonlinearity 104 and differential uniformity 6. As far as we know, such variety of S-boxes, possessing such a good combination of both properties, has not been obtained yet by any other heuristic method, when starting from a random S-box. We compare our results with all other generation methods we know in Table 1. The comparison is with respect to the target properties: nonlinearity, algebraic degree, autocorrelation and differential uniformity. Whenever the value for one of these properties was not reported, we assumed that it had not been considered and we put "-" in the table. Most of the previous works do not consider differential uniformity but only nonlinearity as a target criterion. Therefore, it is hard to compare with their results.

From one side it is to be expected that if one more property is targeted that property will possess better values than if neglected. However, the other properties usually get worse since there are vast connections between those parameters and they cannot be independently optimized. Since most of the previous works do not publish the best S-boxes they have found it is impossible to compare the results. In appendix A we provide one of the best (8×8) S-boxes obtained by *SpImmAlg*.

Table 1: Known methods for generation of (8×8) *bijective* S-boxes

generation methods/properties	$\mathbf{N_S}$	**deg(S)**	$\mathbf{AC(S)_{max}}$	δ**-uniformity**
pseudo-random generation [18, 19]	98	-	-	-
finite field inversion [21]	112	7	32	4
hill climbing method [18]	100	-	-	-
genetic algorithm/hill climbing [19]	100	-	-	-
simulated annealing method [8]	102	-	80	-
special genetic algorithm [27]	104	-	-	-
tweaking method [13]	106	7	56	6
Gradient descent method [16]	104	7	80	8
4-uniform permutations method [22, 23]	98	-	-	4
reversed genetic algorithm [15]	110	7	40	4
reversed genetic algorithm [15]	112	7	32	6
SpImmAlg [this paper]	104	7	88	6

6 Conclusions

The proposed new method produces repeatedly and reasonably fast thousands of *bijective* S-boxes, possessing a very good combination for *nonlinearity* and *differential uniformity* - $(104, 6)$. Such values have only been achieved by finite field inversion-based S-boxes and the methods described in [13,15]. However, these methods either belong to *algebraic constructions*, meaning a limited number of solutions and possible vulnerability to algebraic attacks [9], or by *heuristic algorithms* working in a reverse way, i.e., "skiing-down" from finite field inversion-based S-boxes and obtaining new ones possessing similar or worse properties. The algorithm presented in this paper follows the classical "climbing-up" approach starting from random S-boxes and improving their properties. The results obtained are the best known compared to all previously published works that consider the properties *nonlinearity* and *differential uniformity* and use the same or similar approach [8, 18, 19, 27].

The work presented in this paper can be extended in several directions. The most promising one is to apply some changes in the number of the *mutation* functions and in the functions themselves aiming at producing S-boxes with $N > 104$ and $\delta = 4$ that are different from the finite field inversion-based ones. At least, from [15] we know that such S-boxes exist.

References

1. E. Biham. On Matsui's linear cryptanalysis. In *Eurocrypt'94*, volume 950 of *LNCS*, pages 341–355. Springer, 1994.
2. E. Biham and A. Shamir. Differential cryptanalysis of des-like cryptosystems. In *Advances in Cryptology – CRYPTO'90*, volume 537 of *LNCS*, pages 2–21. Springer Verlag, 1991.

3. E. Biham and A. Shamir. Differential cryptanalysis of des-like cryptosystems. *Journal of Cryptology*, 4:3–72, 1991.
4. J. Brownlee. *Clever Algorithms: Nature-Inspired Programming Recipes*. LuLu, first edition, January 2011.
5. C. Carlet. *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*, chapter Boolean Functions for Cryptography and Error Correcting Codes, pages 257–397. Cambridge University Press, 2010.
6. C. Carlet. *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*, chapter Vectorial Boolean Functions for Cryptography, pages 257–397. Cambridge University Press, 2010.
7. F. Chabaud and S. Vaudenay. Links between differential and linear cryptanalysis. In *Advances in Cryptology – EUROCRYPT'94*, volume 950 of *LNCS*, pages 356–365. Springer Verlag, 1995.
8. J.A. Clark, J.L. Jacob, and S. Stepney. The design of s-boxes by simulated annealing. *New Generation Computing Archive*, 23(3), September 2005.
9. N. T. Courtois and J. Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *Advances in Cryptology - ASIACRYPT'02*, volume 2501 of *LNCS*, pages 267–287. Springer Verlag, 2002.
10. J. Daeman and V. Rijmen. *The design of Rijndael: AES – The advanced Encryption Standard*. Springer Verlag, 2002.
11. J. Daemen, R. Govaerts, and J. Vandewalle. Correlation matrices. In *FSE'94*, volume 1008 of *LNCS*, pages 275–285. Springer, 1995.
12. H. Feistel. Cryptography and computer privacy. *Scientific American*, 228(5):15–23, 1973.
13. J. Fuller and W. Millan. Linear redundancy in s-boxes. In *FSE'03*, volume 2887 of *LNCS*, pages 74–86. Springer, 2003.
14. J. Dj. Golić. Fast low order approximation of cryptographic functions. In *Advances in Cryptology – EUROCRYPT'96*, volume 1070 of *LNCS*, pages 268–282. Springer Verlag, 1996.
15. G. Ivanov, N. Nikolov, and S. Nikova. Reversed genetic algorithms for generation of bijective s-boxes with good cryptographic properties. *IACR Cryptology ePrint Archive (2014), Report 2014/801, http://eprint.iacr.org/2014/801.pdf*.
16. O. Kazymyrov, V. Kazymyrova, and R. Oliynykov. A method for generation of high-nonlinear s-boxes based on gradient descent. *IACR Cryptology ePrint Archive (2013)*.
17. M. Matsui. Linear cryptanalysis method for des cipher. In *Advances in Cryptology – EUROCRYPT'93*, volume 765 of *LNCS*, pages 386–397. Springer Verlag, 1994.
18. W. Millan. How to improve the nonlinearity of bijective s-boxes. In *Australian Conference on Information Security and Privacy 1998*, volume 1438, pages 181–192. Springer Verlag, 1998.
19. W. Millan, L. Burnett, G. Carter, A. Clark, and E. Dawson. Evolutionary heuristics for finding cryptographically strong s-boxes. In *ICICS'99*, volume 1726 of *LNCS*, pages 263–274. Springer, 1999.
20. W. L. Millan. Low order approximation of cipher functions. In *Cryptography: Policy and Algorithms Conference, Proceedings*, volume 1029 of *LNCS*, pages 144–155. Springer Verlag, 1996.
21. K. Nyberg. Differentially uniform mappings for cryptography. In *Advances in Cryptology – EUROCRYPT'93*, volume 765 of *LNCS*, pages 55–64. Springer Verlag, 1994.
22. L. Qu, Y. Tan, C. Li, and G. Gong. More constructions of differentially 4-uniform permutations on $\mathbb{F}_{2^{2k}}$. In *arxiv.org/pdf/1309.7423*, 2013.
23. L. Qu, Y. Tan, C. Tan, and C. Li. Constructing differentially 4-uniform permutations over $\mathbb{F}_{2^{2k}}$ via the switching method. *IEEE Transactions on Inform. Theory*, 59(7):4675–4686, 2013.
24. J. Seberry, X. M. Zhang, and Y. Zheng. Systematic generation of cryptographically robust s-boxes. In *Proceedings of the first ACM Conference on Computer and Communications Security*, pages 171–182. The Association for Computing Machinery, Fairfax, VA, 1993.
25. J. Seberry, X. M. Zhang, and Y. Zheng. Relationships among nonlinearity criteria. In *Advances in Cryptology – EUROCRYPT'94*, volume 950 of *LNCS*, pages 376–388. Springer Verlag, 1995.
26. C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28:656–715, 1949.
27. P. Tesař. A new method for generating high non-linearity s-boxes. *Radioengineering*, 19(1):23–26, 2010.

7 Appendix

Here we present in hex notations the generated by the *SpImmAlg* (8×8) *bijective* S-box S. It was described in Table 1 and has *nonlinearity* 104, *algebraic degree* 7, maximal *autocorrelation* 88 and *differential uniformity* 6:

70 C6 7C D9 97 5D C2 23 D8 CF E7 6E BA D5 88 F2
68 47 25 6C 2B 5B 7A AB 69 B5 C1 8D A6 57 A2 C9
7F FA 67 5C A7 FF 81 8E 09 A4 80 28 14 FE 56 F7
15 13 91 AA AC 3B FC DA 9B 37 D2 46 C3 00 45 AF
10 90 6D B1 D0 5E C7 A1 61 E4 12 F0 F4 38 76 FD
BE E1 59 EB 3F 87 4A 4B E9 54 DB 2A AD D3 29 83
CD 3D 4D DF B0 4E 0E 22 75 F9 03 27 19 8C 3A 1F
B2 66 73 0C 7E 1E 85 8F E2 E8 39 16 94 E0 1C DE
5F 58 7B 44 DD 24 60 95 C5 6A 7D 40 2E EA 64 21
92 F5 26 48 08 B4 01 4C 34 93 79 8B CC 0A B3 98
ED B6 CE 77 63 B8 9D 51 05 F1 11 2C 72 E5 C8 DC
F3 78 4F E3 2D E6 02 9F 18 8A CA CB 82 62 31 2F
41 17 1A BF EC 1B 04 0B 99 32 3E 71 AE 33 A3 53
42 49 D6 BC D4 30 6B A9 FB EE BB 07 EF A5 96 74
5A D1 50 84 43 6F 36 D7 89 A0 65 BD B9 06 C4 9A
A8 3C B7 F8 9E 1D 0F 0D 52 F6 35 86 C0 9C 20 55