information

Article

Generation of Nonlinear Substitutions by Simulated Annealing Algorithm

Alexandr Kuznetsov [1,2], Mikolaj Karpinski [3,4,*], Ruslana Ziubina [3], Sergey Kandiy [2], Emanuele Frontoni [1], Oleksandr Peliukh [2], Olga Veselska [3] and Ruslan Kozak [4]

[1] Department of Political Sciences, Communication and International Relations, University of Macerata, Via Crescimbeni, 30/32, 62100 Macerata, Italy; kuznetsov@karazin.ua (A.K.); emanuele.frontoni@unimc.it (E.F.)
[2] Department of Information and Communication Systems Security, Faculty of Comupter Science, V. N. Karazin Kharkiv National University, 4 Svobody Sq., 61022 Kharkiv, Ukraine; sergeykandy@gmail.com (S.K.); oleksandrpelyukh@gmail.com (O.P.)
[3] Department of Computer Science and Automatics, University of Bielsko-Biala, 2 Willowa St., 43309 Bielsko-Biala, Poland; rziubina@ath.bielsko.pl (R.Z.); oveselska@ath.bielsko.pl (O.V.)
[4] Cyber Security Department, Ternopil Ivan Puluj National Technical University, 56 Ruska St., 46001 Ternopil, Ukraine; ruslank@tntu.edu.ua
* Correspondence: mkarpinski@ath.bielsko.pl

Abstract: The problem of nonlinear substitution generation (S-boxes) is investigated in many related works in symmetric key cryptography. In particular, the strength of symmetric ciphers to linear cryptanalysis is directly related to the nonlinearity of substitution. In addition to being highly nonlinear, S-boxes must be random, i.e., must not contain hidden mathematical constructs that facilitate algebraic cryptanalysis. The generation of such substitutions is a complex combinatorial optimization problem. Probabilistic algorithms are used to solve it, for instance the simulated annealing algorithm, which is well-fitted to a discrete search space. We propose a new cost function based on Walsh–Hadamard spectrum computation, and investigate the search efficiency of S-boxes using a simulated annealing algorithm. For this purpose, we conduct numerous experiments with different input parameters: initial temperature, cooling coefficient, number of internal and external loops. As the results of the research show, applying the new cost function allows for the rapid generation of nonlinear substitutions. To find 8-bit bijective S-boxes with nonlinearity 104, we need about 83,000 iterations. At the same time, the probability of finding the target result is 100%.

Keywords: nonlinear substitution; simulated annealing; S-boxes; combinatorial optimization

Citation: Kuznetsov, A.; Karpinski, M.; Ziubina, R.; Kandiy, S.; Frontoni, E.; Peliukh, O.; Veselska, O.; Kozak, R. Generation of Nonlinear Substitutions by Simulated Annealing Algorithm. *Information* **2023**, *14*, 259. https://doi.org/10.3390/info14050259

Academic Editor: Willy Susilo

Received: 27 March 2023
Revised: 22 April 2023
Accepted: 24 April 2023
Published: 26 April 2023

1. Introduction

Nonlinear substitutions (S-boxes) play an important role in symmetric key cryptography [1–3]. S-boxes fulfil the role of nonlinear complication in converting plaintext into ciphertext, thereby providing resistance to various types of cryptanalysis [4,5]. For instance, the nonlinearity indicator characterises resistance to linear cryptanalysis [6–8]. In this sense, the generation of highly nonlinear substitutions is a relevant field of contemporary research [9–12].

Another important property of S-boxes is the algebraic immunity [13]. This parameter indicates the complexity of the system of algebraic equations describing the substitution [14,15]. Simple systems of equations are derived for algebraic S-boxes, such as in the AES cipher (FIPS-197) [16]. This is considered a potential disadvantage, as the presence of simple mathematical structures can lead to an effective algebraic attack on the cipher [17]. Random substitutions are devoid of hidden algebraic constructions and, in the majority of cases, their algebraic immunity is maximised [18]. However, the nonlinearity of random S-boxes is not high and needs enhancement [19–21].

In order to generate random, highly nonlinear S-boxes, probabilistic combinatorial optimisation algorithms are used [9,10,22]. The generation starts with random substitution.

Further, it is modified by randomly changing the values of the substitution. This procedure is iteratively repeated until the desired (target) result is obtained. At each iteration step, a cost function is calculated, which characterises the degree of proximity to the target result. In other words, the algorithm goes through possible alternatives in a discrete search space, focusing on the values of the cost function.

Among the many probabilistic combinatorial optimisation algorithms, the annealing simulation is worth mentioning [23]. This algorithm has been developed by modelling the physical processes involved in the annealing of metals. As long as the temperature of the metal is high, there is a possibility of atoms transferring to other cells in the crystal lattice. As the temperature drops, this probability decreases and the metal solidifies. This process is modelled as a computational search algorithm in a discrete space of possible states. As long as the temperature is high, there is a probability of transition to a worsening state, i.e., for which the value of the cost function is higher. As the temperature decreases, the probability of adopting a worsening state decreases and the process solidifies—only improving decisions are made. The possibility of making worsening decisions is extremely important, as it enables to move out of the local optimum and continue the search [23].

The annealing simulation algorithm has been used to generate S-boxes in several related works [21,24,25]. However, the complexity of computational search is still very high. For instance, our recent work [25] shows that applying optimised algorithm parameters can achieve the target result in about 450,000 iterations. In this paper we propose a new cost function based on the calculation of the Walsh–Hadamard spectrum. We optimise the annealing simulation algorithm for this cost function and show that the target result can be achieved in about 83,000 iterations. Thus, the main contribution of the paper is a new cost function and optimisation of the simulated annealing algorithm for fast generation of highly nonlinear random substitutions.

2. Related Works

The basic mathematical designations and methods for generating nonlinear substitutions are presented in [4,5]. Random generation methods have been investigated in [19,20]. In [21], one of the first WHS cost functions is proposed and an annealing simulation algorithm is investigated. Other cost functions have been reviewed in [11,26,27]. In particular, the study by Picek, S., Cupic, M., and Rotim, L. [11] proposed a novel cost function that demonstrated significant improvements compared to previous research efforts. This innovative approach contributed to advancing the field of combinatorial optimization for S-box generation, emphasizing the importance of exploring alternative cost functions in achieving better results. In more recent works by Freyre Echevarría, A., and Martínez Díaz [26], another new cost function was introduced, which further enhanced the performance compared to the initial improvements obtained in [11]. This progression highlights the ongoing advancements in the field and the potential for achieving even better results through continuous refinement of cost functions and optimization techniques. Interestingly, the best performance in the studies [26,28] was achieved using the well-established hill climbing algorithm [29,30].

The annealing simulation algorithm has been investigated in several related works. For instance, bijective 8-bit substitutions with nonlinearity 104 were generated in [24]. Nevertheless, the computational complexity of this generation turned out to be very high, requiring more than 3 million iterations. The annealing simulation algorithm was also used in [31], but the authors managed to achieve a nonlinearity of 100, which is significantly lower than other well-known results. In [32] the authors achieved only nonlinearity values of 92, which is comparable to random generation. Our latest work [25] managed to achieve nonlinearity 104 with the lowest generation complexity for the annealing simulation algorithm—we needed about 450,000 iterations.

Another thing to note is the probabilistic nature of substitution generation. The modification of the current state at each iteration of the search is carried out randomly Consequently, the target result is not always achieved. As an example, in [19,21] the probability

of generating S-boxes with nonlinearity 102 is about 0.5%. More than 50% of the algorithm's runs yielded the target result in our work [25]. In this paper, we significantly improve on these two important indicators—the complexity of generation and the probability of achieving the target outcome.

3. Background

The annealing simulation algorithm is based on the simulation of the physical process that occurs during the crystallisation of a substance, including metals. At high temperature, the atoms of the substance can move to other cells of the crystal lattice and this is simulated by means of the so-called Gibbs measure [33]. The probability of the system X being in state x (equivalently, that the random variable X has value of x) is defined as (Gibbs measure):

$$P(X = x) = \frac{e^{-\beta E(x)}}{Z(\beta)},$$

where $E(x)$ s interpreted as state energy of x; $\beta = \frac{1}{k_B T}$ is thermodynamic beta; T is the temperature; k_B is Boltzmann constant; $Z(\beta)$ is the normalizing constant.

The annealing simulation algorithm simulates a similar process. The state energy $E(x)$ is interpreted as the value of the cost function $CF(S)$ at some point in the search space. In our case, S is a particular S-box in the discrete space of all possible substitutions. By modifying the state S, we get a new state S^*. In doing so, there are possible cases:

- The energy has reduced, i.e., $CF(S^*) < CF(S)$. Then the substitution S^* is taken as the current state and the search continues from this point in the state space; and
- The energy has not reduced, i.e., $CF(S^*) \geq CF(S)$. Then the substitution S^* is taken as the current state with probability.

$$P(X = S^*) = e^{-\frac{CF(S^*) - CF(S)}{T}}.$$

Thus, the probability of adopting a deteriorating state diminishes as the temperature T decreases. The algorithm simulates the cooling of the substance in proportion to the cooling coefficient α.

The pseudocode of the annealing simulation algorithm has the form [21,23,34]:

Algorithm 1. The Annealing Simulation Algorithm

Input : $T_0; \alpha; S_0$.
$T \leftarrow T_0$;
$S \leftarrow S_0$;
For $i = 1$ to k_{out}:

- $S^* \leftarrow$ Modification(S);
- If $[CF(S^*) < CF(S)] \vee [RND(0,1) < P(X = S^*)]$
  - $S \leftarrow S^*$;
- $T \leftarrow \alpha T$;

Output: the final state S.

In [25] we refined this algorithm, in particular we changed the new state acceptance rule:

$$[N(S^*) > N(S)] \vee [CF(S^*) \leq CF(S)] \vee [RND(0,1) < P(X = S^*)],$$

i.e., we accept the state S^* when:

- Either the nonlinearity has increased, $N(S^*) > N(S)$; or
- The value of the cost function has not increased, $CF(S^*) \leq CF(S)$; or
- With probability of $P(X = S^*)$ we accept a worsening solution.

This change has greatly extended the range of accepted values. In addition, as in [35], we use:

- An additional loop of k_{int} iterations at the same temperature T; and
- An additional condition for exiting the outer loop when the number of iterations without improvement k_{froz} is reached.

4. Materials and Methods

We investigate the problem of generating bijective 8-bit substitutions given by a set of eight Boolean functions $S(x) = (f_0, f_1, \ldots, f_7)$ from eight Boolean variables:

$$f_0(x_0, x_1, \ldots, x_7), f_1(x_0, x_1, \ldots, x_7), \ldots, f_7(x_0, x_1, \ldots, x_7).$$

The nonlinearity of the substitution S is defined through the Walsh–Hadamard spectrum by the expression [5]:

$$N(S) = \min_{v \in \{0,1\}^8 \setminus \{0\}^8} \{N(v \cdot S(x))\} = \frac{1}{2}(2^8 - \max_{v,u \in \{0,1\}^8 \setminus \{0\}^8} |WHT(v \cdot S(x), u)|),$$

where

$$WHT(f(x), u) = \sum_{x \in \{0,1\}^8} (-1)^{f(x) \oplus u \cdot x}.$$

Obviously, to increase the nonlinearity of $N(S)$, the maximum absolute values of the Walsh–Hadamard coefficients must be reduced. In the combinatorial optimisation algorithms under research, this is implemented through the cost function $CF(S)$—an interpretation of the energy of state $E(x)$ during the crystallisation of matter.

One of the best known cost functions used to generate nonlinear substitution is proposed in [21]:

$$CF_{WHS}(S) = \sum_{v \in \{0,1\}^8} \sum_{u \in \{0,1\}^8} ||WHT(v \cdot F(x), u)| - X|^R,$$

where X and R—configurable parameters.

The $CF_{WHS}(S)$ function has been used in various related works. For instance, in [11,28,36] it was used to find substitutions with nonlinearity $N(S) = 104$. However, the complexity of the search turned out to be extremely high, requiring at least 3 million iterations to find a single S-box. In addition, the probability of achieving the target result did not exceed 1/200 = 0.5% [21]. Another cost function was proposed in [11]:

$$CF_{PCF}(S) = \sum_{i=1}^{N} 2^{-i} H(S)_{k-i},$$

where $H(S)$—vector of values $|WHT(v \cdot F(x), u)|$, in which the i-th position shows the number of coefficients with values $|4i|$, k—maximum position number with a non-zero value.

In [11], S-boxes with nonlinearity $N(S) = 104$ were generated using $CF_{PCF}(S)$. The authors of this paper investigated various probabilistic search algorithms: Genetic Algorithm (GA) [37], Genetic and Tree Algorithm (GaT) [36], and our Local Search Algorithm (LSA). The combination of $CF_{PCF}(S)$ and GaT turned out to be the most effective [11].

In [26,28] a function was suggested:

$$CF_{WCF}(S) = \sum_{v \in \{0,1\}^8} \sum_{u \in \{0,1\}^8} \prod_{z \in C} ||WHT(v \cdot F(x), u)| - z|,$$

where $C = \{0, 4, \ldots, 32\}$.

The authors of these papers also investigated LSA and GaT, however, the combination of the $CF_{WCF}(S)$ cost function and Hill Climbing (HC) Algorithm proved to be the most

effective for forming bijective S-boxes with $N(S) = 104$ [26]. However, the success rate was only 11/30, i.e., only about 35% of the algorithm runs ended up generating a target S-box with $N(S) = 104$ [26].

In this paper we propose a new cost function that is based on a Walsh–Hadamard spectrum calculation:

$$CF_{new} = \sum_{\substack{v\in\{0,1\}^8 \\ |WHT(v\cdot F(x),u)|>40}} \sum_{u\in\{0,1\}^8} 2^{\frac{|WHT(v\cdot F(x),u)| \ - \ 20}{2}}.$$

The calculation of our cost function is done through the summation of the $2^{\frac{|WHT(v\cdot F(x),u)| \ - \ 20}{2}}$ terms, for which $|WHT(v \cdot F(x), u)| > 40$. This allows focusing only on those $WHT(f(x), u)$ coefficients that can change the nonlinearity value of $N(S)$. Such logic for forming the cost function will, in our view, significantly reduce the computational complexity of the state space search.

The aim of our work was to investigate the behaviour of the annealing simulation algorithm with the new CF_{new} cost function under different input data. In particular, we examined the different values:

- The initial temperature T_0;
- The cooling rate α; and
- The number of internal k_{int} and external k_{out} loops.

The selection of the initial temperature is important, as it is decisive for calculating the probability of accepting a deteriorating condition $P(X = S^*)$. We investigated four initial temperature values $T_0 \in \{100, 1000, 10,000, 100,000\}$. Figure 1 shows the calculated dependencies of the probability $P(X = S^*)$ on the value of $\Delta = CF(S^*) - CF(S)$ for different values of temperature T.

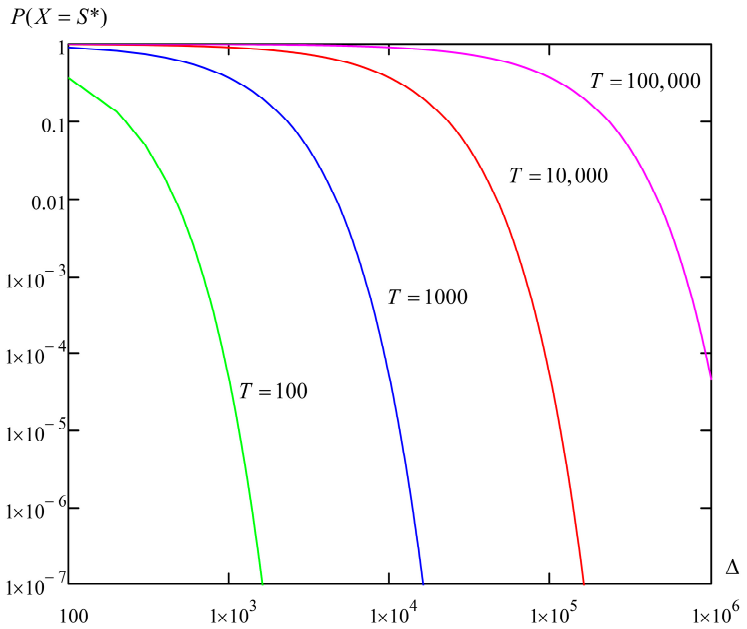

Figure 1. Dependencies of the probability of accepting deteriorating states on the value of $\Delta = CF(S^*) - CF(S)$.

The probability of accepting deteriorating conditions should not be excessively high and should not be too low. For example, if $P(X = S^*) \approx 1$, then almost every deteriorating condition will be accepted. In this case, the algorithm does not search for the optimum state, but only goes through possible alternatives. Conversely, if $P(X = S^*) \approx 0$ deteriorating decisions will not be made at all and the algorithm will only find the nearest local minimum

of the cost function. As can be seen from Figure 1, all probability values shift to the right as T increases, i.e., for each value of T there is a range of "acceptable" $\Delta = CF(S^*) - CF(S)$ values for which the probabilities $P(X = S^*)$ are neither too large nor too small.

It should also be noted that the temperature decreases at each iteration of the algorithm in proportion to the cooling factor α. This leads to a reduction in the probability $P(X = S^*)$, as illustrated in Figure 2 for $T_0 = 1000$ and $\Delta = CF(S^*) - CF(S) = 1000$. In our research, we considered $\alpha \in \{0.1; 0.2; \ldots; 0.9\}$ values and independently tested the search algorithm for each one.

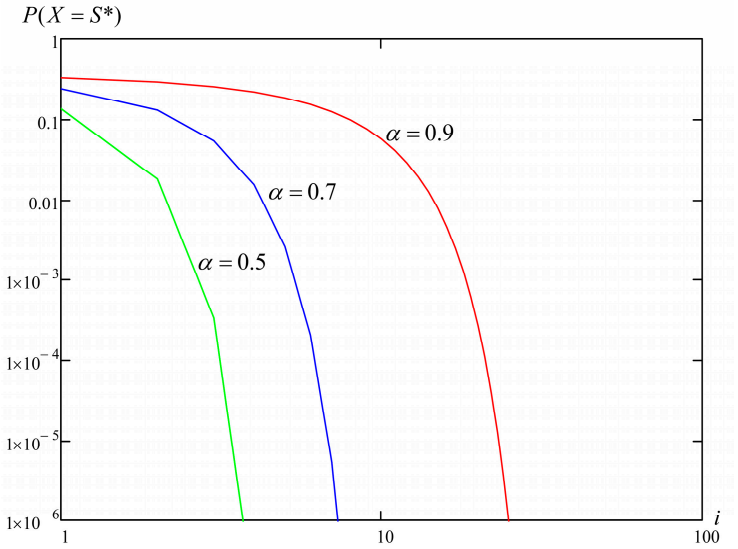

Figure 2. Dependencies of the probability of accepting deteriorating states on the external loop iteration number, $T_0 = 1000$, $\Delta = CF(S^*) - CF(S) = 1000$.

At the same time, we chose the number of internal k_{int} and external k_{out} loops so that the total number of iterations (in one thread) would be equal to $k_{out} \cdot k_{\text{int}} = 100,000$. We used multi-threaded mode, testing was performed on 8 threads, i.e., the total number of iterations for one algorithm run did not exceed 800,000.

In total, we considered three options for starting the algorithm:

- $k_{out} = 10$, $k_{\text{int}} = 10{,}000$;
- $k_{out} = 100$, $k_{\text{int}} = 1000$; and
- $k_{out} = 1000$, $k_{\text{int}} = 100$.

The first option corresponds to a very slow decline in temperature and it means that worsening conditions will be taken more often. The second and third options use a greater number of external iterations and this means that the probability of accepting a deteriorating condition is reduced rapidly.

We investigated both the total number of search iterations and the probability of finding the target state (an 8-bit bijective S-box with $N(S) = 104$ and maximum algebraic immunity).

5. Results

The main results are shown in Tables 1–4.

Table 1. Findings of search cost investigations for $T_0 = 100$.

	k_{out} = 10, k_{int} = 10,000	k_{out} = 100, k_{int} = 1000	k_{out} = 1000, k_{int} = 100
$\alpha = 0.1$	76,009.67 (100%)	79,524.40 (100%)	80,626.22 (100%)
$\alpha = 0.2$	81,687.52 (100%)	90,611.95 (100%)	78,434.70 (100%)
$\alpha = 0.3$	81,329.86 (100%)	88,245.98 (100%)	85,446.05 (100%)

Table 1. *Cont.*

	k_{out} = 10, k_{int} = 10,000	k_{out} = 100, k_{int} = 1000	k_{out} = 1000, k_{int} = 100
$\alpha = 0.4$	80,237.73 (100%)	72,907.94 (100%)	90,480.17 (100%)
$\alpha = 0.5$	92,201.78 (100%)	82,822.92 (100%)	85,577.41 (100%)
$\alpha = 0.6$	85,267.11 (100%)	80,468.54 (100%)	83,661.40 (100%)
$\alpha = 0.7$	86,358.69 (100%)	79,653.47 (100%)	83,008.08 (100%)
$\alpha = 0.8$	77,112.92 (100%)	82,100.52 (100%)	85,446.05 (100%)
$\alpha = 0.9$	80,084.09 (100%)	84,360.09 (100%)	80,019.55 (100%)

Table 2. Findings of search cost investigations for $T_0 = 1000$.

	k_{out} = 10, k_{int} = 10,000	k_{out} = 100, k_{int} = 1000	k_{out} = 1000, k_{int} = 100
$\alpha = 0.1$	82,530.71 (100%)	84,092.05 (100%)	82,686.53 (100%)
$\alpha = 0.2$	83,479.26 (100%)	81,281.64 (100%)	83,755.94 (100%)
$\alpha = 0.3$	84,975.59 (100%)	79,385.64 (100%)	88,544.81 (100%)
$\alpha = 0.4$	82,686.68 (100%)	83,480.77 (100%)	93,185.31 (100%)
$\alpha = 0.5$	82,397.30 (100%)	81,515.59 (100%)	80,625.44 (100%)
$\alpha = 0.6$	85,553.65 (100%)	78,870.06 (100%)	82,309.30 (100%)
$\alpha = 0.7$	83,126.02 (100%)	89,313.00 (100%)	86,520.43 (100%)
$\alpha = 0.8$	78,633.10 (100%)	82,243.44 (100%)	81,308.54 (100%)
$\alpha = 0.9$	86,604.24 (100%)	90,141.24 (100%)	83,131.90 (100%)

Table 3. Findings of search cost investigations for $T_0 = 10,000$.

	k_{out} = 10, k_{int} = 10,000	k_{out} = 100, k_{int} = 1000	k_{out} = 1000, k_{int} = 100
$\alpha = 0.1$	81,975.30 (100%)	79,378.70 (100%)	83,423.39 (100%)
$\alpha = 0.2$	88,910.29 (100%)	88,009.86 (100%)	83,627.32 (100%)
$\alpha = 0.3$	84,952.69 (100%)	83,338.59 (100%)	77,009.26 (100%)
$\alpha = 0.4$	80,722.98 (100%)	82,152.62 (100%)	90,112.30 (100%)
$\alpha = 0.5$	75,690.74 (100%)	83,605.75 (100%)	84,112.64 (100%)
$\alpha = 0.6$	82,985.41 (100%)	82,453.39 (100%)	80,729.33 (100%)
$\alpha = 0.7$	86,407.67 (100%)	82,462.94 (100%)	84,714.74 (100%)
$\alpha = 0.8$	82,742.78 (100%)	80,918.05 (100%)	86,715.16 (100%)
$\alpha = 0.9$	81,975.30 (100%)	79,378.70 (100%)	83,423.39 (100%)

Table 4. Findings of search cost investigations for $T_0 = 100,000$.

	k_{out} = 10, k_{int} = 10,000	k_{out} = 100, k_{int} = 1000	k_{out} = 1000, k_{int} = 100
$\alpha = 0.1$	129,340.79 (100%)	82,815.54 (100%)	84,054.20 (100%)
$\alpha = 0.2$	153,047.44 (100%)	88,950.34 (100%)	80,751.58 (100%)
$\alpha = 0.3$	177,362.26 (100%)	83,268.43 (100%)	79,454.68 (100%)
$\alpha = 0.4$	185,580.21 (100%)	86,333.64 (100%)	82,799.83 (100%)
$\alpha = 0.5$	226,627.30 (100%)	92,578.85 (100%)	91,702.98 (100%)
$\alpha = 0.6$	281,077.52 (100%)	87,348.77 (100%)	80,655.30 (100%)
$\alpha = 0.7$	345,715.50 (100%)	94,778.97 (100%)	83,609.55 (100%)
$\alpha = 0.8$	503,914.17 (100%)	106,267.47 (100%)	79,737.76 (100%)
$\alpha = 0.9$	- (0%)	153,180.84 (100%)	81,359.10 (100%)

Table 1 shows the results of the search cost estimate (average number of iterations over all threads) for $T_0 = 100$. It is observed for all cases that the target result is consistently generated in approximately 83,000 iterations. The search cost is not affected by either the cooling factor or the number of iterations of the outer k_{out} and inner k_{int} loops of the algorithm. This suggests that the proportion of deteriorating decisions taken is not large

and does not contribute significantly to the overall cost of the search. The same conclusions can be drawn from the results shown in Tables 2 and 3 for $T_0 = 1000$ and $T_0 = 10,000$. In brackets is the fraction of runs of the algorithm in which the target state was found, i.e., an 8-bit bijective S-box with $N(S) = 104$ accompanied by maximum algebraic immunity was generated. As can be seen from the results shown in Tables 1–3, 100% of the algorithm runs ended up generating the target S-box in all cases.

The initial temperature $T_0 = 100,000$ gives slightly different results. As can be seen from the results in Table 4, the cost of searching for some parameters significantly increased. For instance, for $k_{out} = 100$, $k_{\text{int}} = 1000$ and $\alpha = 0.9$ the simulated annealing algorithm needs about twice as many iterations to find the target result (compared to Tables 1–3). The situation worsens for $k_{out} = 10$, $k_{\text{int}} = 10,000$ and at $\alpha = 0.9$ no target result was found (over 800,000 search iterations in each of the 100 runs of the algorithm). The increase in search costs is highlighted in colour in Table 4. The darkest cell colour means the worst result.

6. Discussion of the Results

The stable formation of the target outcome for all parameters in Tables 1–3 is due to the low probability of acceptance of a deteriorating condition. For example, Appendix A shows a fragment of the list of cost function changes with the nonlinearity, current temperature and iteration number for $T_0 = 1000$. The algorithm did not take a worsening condition even once, since for even the smallest observable $\Delta = CF(S^*) - CF(S) \approx 10,000$ the corresponding probability is extremely low $P(X = S^*) \approx 4.5{\cdot}10^{-5}$. Nevertheless, deteriorating conditions are accepted, but under a different rule. As a reminder, our version of the algorithm accepts all states for which $N(S^*) > N(S)$, even if $CF(S^*) > CF(S)$. In other words, our algorithm actually uses two cost functions: $CF(S)$ and $N(S)$. This enables the algorithm to take states that degrade the $CF(S)$ function and thus escape from the local optimum even at very low values of probability $P(X = S^*)$. This can be clearly seen in the example given in Appendix A, with the relevant cases in bold. Appendixes also show the resulting S-boxes and indicate the main cryptographic indicators: nonlinearity (NL), delta uniformity (DU) and algebraic immunity (AI).

As the initial temperature rises, the proportion of accepted worsening conditions increases. This is explained by the raising of the probability of $P(X = S^*)$. For instance, for $T_0 = 10,000$ and $\Delta = CF(S^*) - CF(S) \approx 10,000$ we obtain $P(X = S^*) \approx 0.37$. Appendix B contains a fragment of the list of changes in the cost function for this initial temperature, with the case of accepting a worsening condition highlighted in bold.

It should be noted that almost identical results were obtained for the initial temperature $T_0 \in \{100,\ 1000,\ 10,000\}$. The explanation for this is obviously the compensation of taking possible states due to the $[N(S^*) > N(S)] \vee [CF(S^*) \leq CF(S)] \vee [RND(0,1) < P(X = S^*)]$ rule. In fact, we observe a stable operation of the probabilistic search algorithm over a wide range of input data.

Appendix C shows a fragment of the list of cost function changes for $T_0 = 100,000$. In this case, the chances of accepting deteriorating conditions are unsustainably high. For example, for $\Delta = CF(S^*) - CF(S) \approx 10,000$ we obtain $P(X = S^*) \approx 0.9$, i.e., almost every deteriorating condition is accepted and this significantly increases the complexity of the search. The relevant state acceptances are highlighted in Appendix C.

Separately, it should be commented on the high probability of forming a target S-box with $N(S) = 104$. In almost every case, we observed a 100% achievement of the desired result. In the single, worst case, for the input parameters: $T_0 = 100,000$, $\alpha = 0.9$, $k_{out} = 10$, $k_{\text{int}} = 10,000$ (highlighted with the darkest background in Table 4) the probability was zero, i.e., the algorithm could not find any target results for these parameters.

To explain this observation, we conducted additional research. We plotted histograms of the number of iterations for the empirical data under the following parameters (Figures 3 and 4):

- Case 1: T_0 = 100,000, α = 0.9, k_{out} = 1000, k_{int} = 100 (the mean is 81,359.10; the standard deviation is 28,406.44); and
- Case 2: T_0 = 100,000, α = 0.7, k_{out} = 10, k_{int} = 10,000 (the mean is 345,715.50; the standard deviation is 37,060.07).

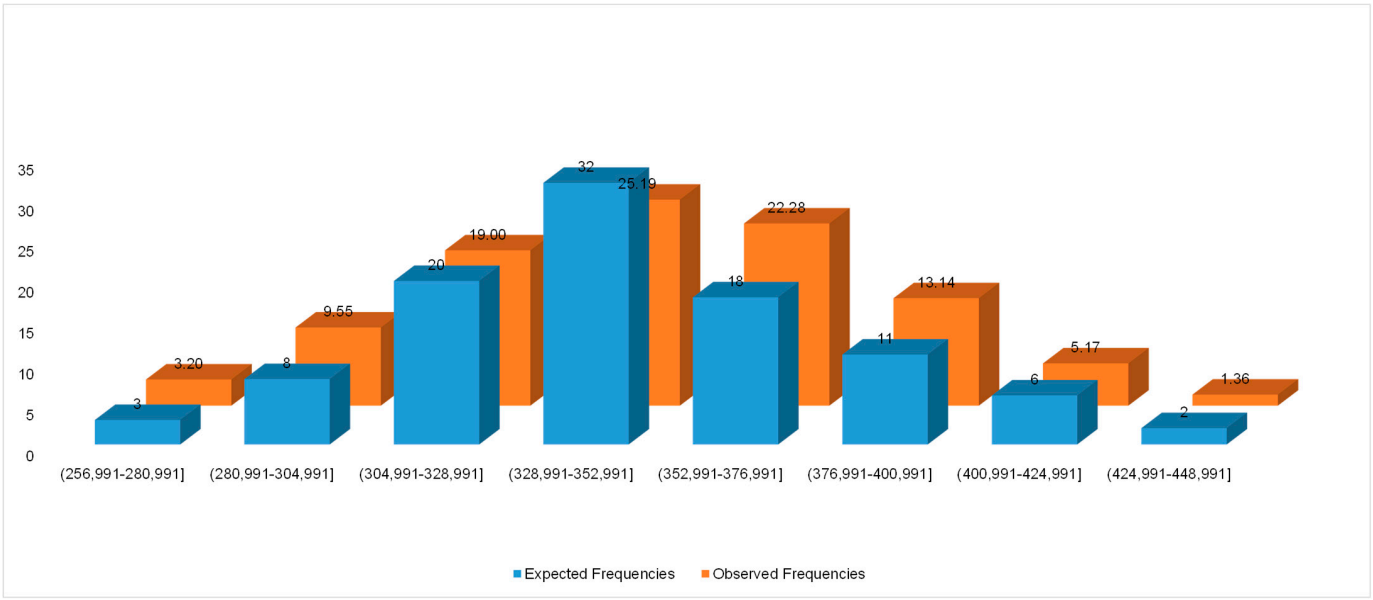

Figure 3. Expected (theoretical) and observed (empirical) frequencies for T_0 = 100,000, α = 0.9, k_{out} = 1000, k_{int} = 100 (the mean is 81,359.10; the standard deviation is 28,406.44).

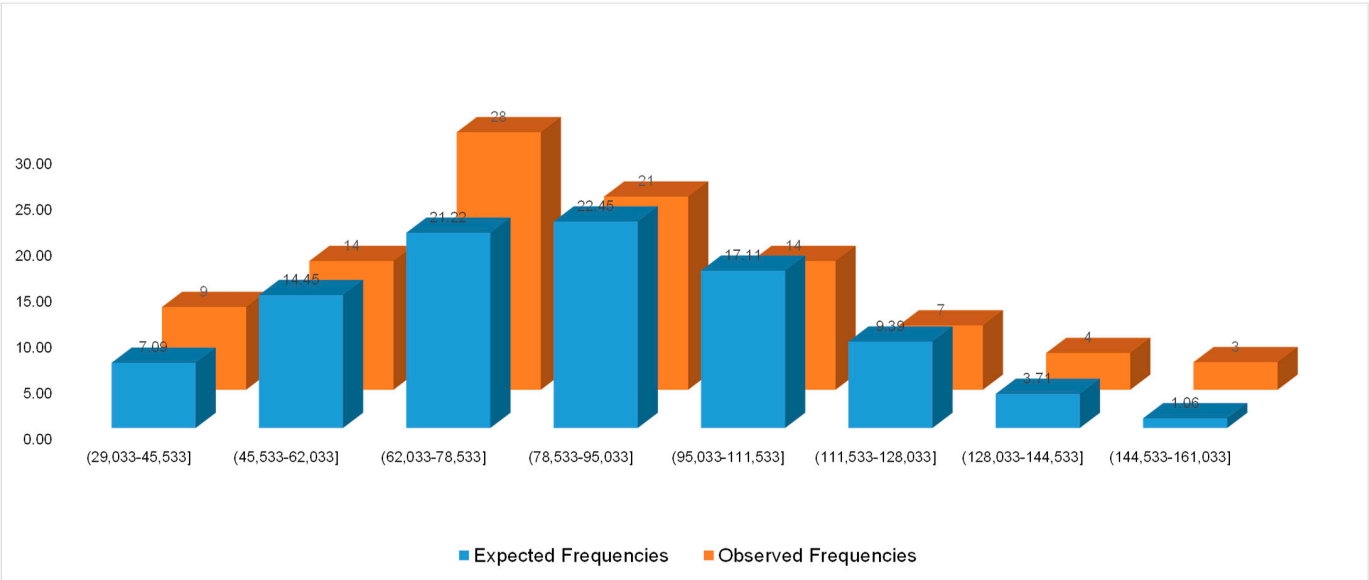

Figure 4. Expected (theoretical) and observed (empirical) frequencies for T_0 = 100,000, α = 0.7, k_{out} = 10, k_{int} = 10,000 (the mean is 345,715.50; the standard deviation is 37,060.07).

The figures also show the expected frequencies for a normal distribution.

Tables 5 and 6 show the results of testing the normal distribution hypothesis. In both cases, the calculated value of the χ^2 does not fall into the critical region, hence the hypothesis is confirmed. This means that there is a 95% probability that the number of iterations of the annealing simulation algorithm is distributed according to a normal distribution law.

Table 5. Checking the hypothesis of a normal distribution for T_0 = 100,000, α = 0.9, k_{out} = 1000, k_{int} = 100 (the mean is 81,359.10; the standard deviation is 28,406.44).

Value Interval	Expected Frequencies m_i (Normal Distribution)	Observed Frequencies x_i (Empirical Data)	$\frac{(x_i - m_i)^2}{m_i}$
(29,033–45,533]	7.09	9	0.52
(45,533–62,033]	14.45	14	0.01
(62,033–78,533]	21.22	28	2.16
(78,533–95,033]	22.45	21	0.09
(95,033–111,533]	17.11	14	0.56
(111,533–128,033]	9.39	7	0.61
(128,033–144,533]	3.71	4	0.02
(144,533–161,033]	1.06	3	3.58
Total $k = 8$ intervals			$\chi^2 = \sum_{i=1}^{k} \frac{(x_i - m_i)^2}{m_i} = 7.56$
		The significance level is $\alpha = 0.05$. The critical region is >11.070498	

Table 6. Checking the hypothesis of a normal distribution for T_0 = 100,000, α = 0.7, k_{out} = 10, k_{int} = 10,000 (the mean is 345,715.50; the standard deviation is 37,060.07).

Value interval	Expected Frequencies m_i (Normal Distribution)	Observed Frequencies x_i (Empirical Data)	$\frac{(x_i - m_i)^2}{m_i}$
(256,991–280,991]	3.20	3	0.01
(280,991–304,991]	9.55	8	0.25
(304,991–328,991]	19.00	20	0.05
(328,991–352,991]	25.19	32	1.84
(352,991–376,991]	22.28	18	0.82
(376,991–400,991]	13.14	11	0.35
(400,991–424,991]	5.17	6	0.13
(424,991–448,991]	1.36	2	0.31
Total $k = 8$ intervals			$\chi^2 = \sum_{i=1}^{k} \frac{(x_i - m_i)^2}{m_i} = 3.77$
		The significance level is $\alpha = 0.05$. The critical region is >11.070498	

Figure 5 shows graphs of the normal probability distributions for these two cases. The area under each curve is equal to 1. The total number of iterations in our research was set at 800,000 (100,000 for each of the 8 threads). Consequently, the area under the part of the curve for the values (border highlighted with a dashed line) equals the probability of the search algorithm not finding the target result. As can be seen from the graphs for both Case 1 and Case 2, this probability tends towards zero. This explains the 100% finding of the target states for these parameters. We repeated these examinations for each case in Tables 1–4 and our theoretical calculations fully coincide with these empirical observations.

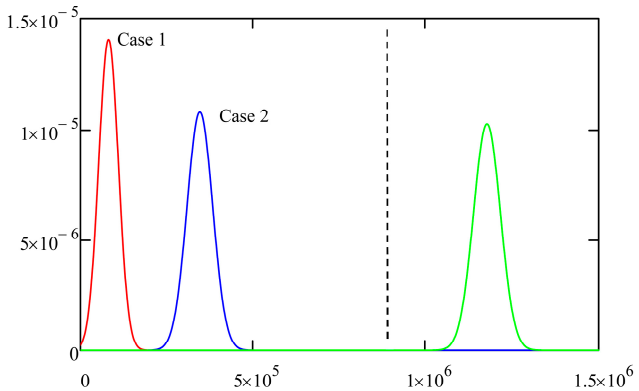

Figure 5. Plots of the normal probability distribution.

Additionally, we conducted experiments for the worst-case case (highlighted in darker colour in Table 4). To do this, we increased the number of external loops by a factor of 10 and performed several test runs of the algorithm. For the values of the mean and standard deviation obtained, Figure 5 shows a graph of the normal distribution (far right). For this graph, the area under the part of the curve for the values tends to 1, which explains the zero probability of forming the target outcome in Table 4.

Further discussion of the obtained results, particularly in the context of the comparison results presented in Table 7, reveals the significance of our study in advancing the field of combinatorial optimization for S-box generation. The improvements achieved by our approach are evident when contrasted with the outcomes of other known works.

Table 7. Comparison of results.

	[21,38]	[32]	[24]	[25]	**Our Paper**
Nonlinearity of substitution $N(S)$	102	92	104	104	104
Probability of generating target S-boxes	0.5%	–	–	56.4%	100%
Search cost (average number of iterations)	–	–	3,000,000	450,000	83,000

Table 7 provides a comprehensive comparison of our approach with other methods for generating nonlinear S-boxes, highlighting the substantial reduction in the average number of algorithm search iterations. This reduction by more than eight times demonstrates the effectiveness of our custom implementation of the simulated annealing algorithm and the novel cost function. Moreover, it suggests that our method is more efficient and computationally less expensive than previously proposed solutions.

Another notable result in Table 7 is the enhanced frequency of reaching the target result. By employing probabilistic methods for generating substitutions, we have managed to increase the success rate by nearly two-fold. This improvement indicates that our approach is not only faster but also more reliable and robust in generating high-quality S-boxes, which is crucial for cryptographic applications.

The normal distribution of the search algorithm's iteration count, as confirmed by the histograms, provides additional insight into the consistency and predictability of our method. The alignment between theoretical calculations and experimental results further supports the adequacy of our approach.

In conclusion, the results presented in Table 7, along with the other findings in our study, emphasize the advantages of our method in terms of efficiency, success rate, and reliability. These improvements have the potential to impact the development of more secure cryptographic algorithms and contribute to the ongoing research in the field of combinatorial optimization. Future work can further explore the potential of other probabilistic search algorithms and examine their effectiveness in conjunction with the novel cost function.

7. Conclusions

In this study, we investigated the simulated annealing algorithm for solving a combinatorial optimization problem related to the search for nonlinear substitutions (S-boxes). S-boxes play a crucial role in ensuring the efficiency of modern symmetric key cryptographic algorithms. We generated 8-bit bijective S-boxes with a nonlinearity of $N(S) = 104$, utilizing our custom implementation of the simulated annealing algorithm and a novel cost function.

The experimental results demonstrate that, in most cases, the formation of target S-boxes is achieved within a stable average of 83,000 iterations. This performance ranks among the best-known results for the simulated annealing algorithm. Table 7 provides comparisons with other known works, illustrating that our approach significantly improves upon existing results. The average number of algorithm search iterations has been reduced by more than 8 times.

We also investigated the distribution of the search algorithm's iteration count. Histograms of iteration distributions were constructed, confirming our hypothesis of a normal distribution with a 95% probability. This theoretical distribution enables the calculation of the probability of forming the target result for each considered case. The theoretical calculations align with experimental results, substantiating the adequacy of the obtained outcomes.

Another improvement in S-box generation achieved in our study is the frequency of reaching the target result. By exploring probabilistic methods of generating substitutions, we managed to nearly double the success rate.

Thus, we have significantly advanced the state-of-the-art in this research field in terms of these two crucial metrics. This progress is attributed to the utilization of the novel cost function and the optimized implementation of the simulated annealing algorithm.

A promising direction for future research involves the implementation of other probabilistic search algorithms, such as GA, GaT, and LSA, and testing their efficiency using the new cost function.

Author Contributions: Conceptualization and methodology, A.K., M.K. and R.Z.; software and validation, S.K.; formal analysis, investigation, E.F. and O.V.; writing—review and editing, O.P. and R.K. All authors have read and agreed to the published version of the manuscript.

Funding: The research work reported in this paper was in part supported by the National Centre for Research and Development, Poland under the project No. POIR.04.01.04-00-0048/20.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Starting simulated annealing . . .
Parameters:
Thread count: 8
Max outer loops in thread: 10
Max inner loops in thread: 10,000
Initial temperature: 1000
Alpha parameter: 0.9
Max frozen loops: 1,000,000
cost = 1.56004×10^8 NL = 92 temperature = 1000 Iteration = 2
cost = 1.47575×10^8 NL = 92 temperature = 1000 Iteration = 4
. . .
cost = 1.27631×10^7 NL = 100 temperature = 1000 Iteration = 218
cost = 1.25747×10^7 NL = 98 temperature = 1000 Iteration = 232
cost = 1.38404×10^7 NL = 100 temperature = 1000 Iteration = 233
cost = 1.36724×10^7 NL = 98 temperature = 1000 Iteration = 235
cost = 1.36888×10^7 NL = 100 temperature = 1000 Iteration = 236
cost = 1.32383×10^7 NL = 98 temperature = 1000 Iteration = 239
cost = 1.38281×10^7 NL = 100 temperature = 1000 Iteration = 243
cost = 1.37953×10^7 NL = 100 temperature = 1000 Iteration = 245
cost = 1.23986×10^7 NL = 98 temperature = 1000 Iteration = 249
cost = 1.21815×10^7 NL = 98 temperature = 1000 Iteration = 260
cost = 1.19685×10^7 NL = 98 temperature = 1000 Iteration = 271
cost = 1.24805×10^7 NL = 100 temperature = 1000 Iteration = 280
cost = 1.23986×10^7 NL = 100 temperature = 1000 Iteration = 291
. . .

cost = 2.72794 $\times$ 10^6 NL = 102 temperature = 1000 Iteration = 54,480
cost = 2.69926 $\times$ 10^6 NL = 102 temperature = 1000 Iteration = 54,801
cost = 2.62144 $\times$ 10^6 NL = 102 temperature = 1000 Iteration = 78,822
cost = 2.62144 $\times$ 10^6 NL = 102 temperature = 900 Iteration = 86,252
SEARCH COST: 86,253
target sbox:
D3, 8E, E7, 8A, 12, AD, 72, 26, 81, 70, 36, 9D, 19, 37, EE, CF, FD, 38, 6D, 1C, F6, 5D, 93, BD, 68, EB, 5C, 77, 2F, 96, 01, 2C, A4, 41, 73, 07, 40, 71, F0, 2B, C1, 4B, 28, E3, AA, D9, 4C, 53, 0E, BE, B3, 1D, 43, 3E, E8, DF, E4, 9B, BB, 2E, 0A, 57, 85, 7A, 1A, E1, 47, 56, C0, ED, B1, F2, DA, AE, E0, 7D, 98, D5, 0F, AC, 25, DD, 67, 33, DC, 65, 0D, D6, DE, A6, F5, 4A, 8F, 2A, 6F, EF, 86, B9, 6E, 6C, C6, 42, 89, 39, F1, 88, A0, A8, 13, EC, 95, A7, C8, 7E, 27, A5, 7B, BA, F8, 22, 3B, 05, 90, 21, AB, 1B, 3A, F7, A9, 60, 50, 10, 18, FA, CC, 04, 5F, 8C, 78, 5A, E2, 94, 0C, 59, 9A, BF, D2, 08, 35, 46, 99, E5, 30, 82, F9, 92, 6A, 1E, 63, 49, E9, 2D, CB, DB, 7F, D8, 4E, F4, 87, 74, C9, 34, 97, 15, F3, 09, 52, 79, 7C, 45, 3C, C5, 03, 44, 58, 31, B4, 8D, 64, 8B, 3D, 3F, EA, E6, D7, D1, 54, 4D, 84, 5E, 69, 0B, 02, 20, 9E, 06, FC, A2, CE, 83, 80, AF, CD, A3, 62, 61, A1, 11, 4F, B6, 17, 5B, C7, 16, C4, BC, C3, B7, 55, 51, 29, 48, B8, 1F, 00, 6B, 24, 32, B0, 75, 9F, D4, C2, B2, 91, CA, FB, 23, B5, 76, 66, 9C, D0, FF, 14, FE,
NL = 104
DU = 10
AI = 3

Appendix B

Starting simulated annealing . . .
Parameters:
Thread count: 8
Max outer loops in thread: 100
Max inner loops in thread: 1000
Initial temperature: 10,000
Alpha parameter: 0.5
Max frozen loops: 1,000,000
target NL: 104
cost = 1.19575 $\times$ 10^8 NL = 92 temperature = 10,000 Iteration = 8
cost = 7.38345 $\times$ 10^7 NL = 94 temperature = 10,000 Iteration = 9
. . .
cost = 2.78036 $\times$ 10^7 NL = 96 temperature = 10,000 Iteration = 95
cost = 2.78077 $\times$ 10^7 NL = 96 temperature = 10,000 Iteration = 97
cost = 2.33759 $\times$ 10^7 NL = 98 temperature = 10,000 Iteration = 99
. . .
cost = 2.95322 $\times$ 10^6 NL = 102 temperature = 312.5 Iteration = 41,633
cost = 2.95322 $\times$ 10^6 NL = 102 temperature = 312.5 Iteration = 43,285
. . .
cost = 2.39206 $\times$ 10^6 NL = 102 temperature = 0.0762939 Iteration = 143,542
cost = 2.39206 $\times$ 10^6 NL = 102 temperature = 0.038147 Iteration = 144,064
SEARCH COST: 144,064
target sbox:
31, 7D, B3, 1A, 69, 28, D3, 86, 79, 14, FB, CC, 38, 25, 5C, 3E, C7, DD, 00, 5A, 5D, 97, A7, 62, F7, D9, 60, 44, AB, AC, B6, EC, 3B, DF, 2D, 89, CD, 59, 7E, 13, B9, 78, 2E, BB, EE, A6, 7F, 85, B8, 40, D0, 4F, 30, 9C, 70, 7A, 77, 21, 32, CA, E7, E1, 15, 7B, A1, FA, BA, DA, F4, 3D, 66, 9D, 76, 3C, 84, EB, 54, E8, 26, 96, 68, 63, A4, B1, 53, 6F, 29, 8C, E5, 5F, BD, AA, 6C, A2, F0, 51, 95, 35, FE, 05, EA, F3, 7C, FF, 3A, 10, E4, 0C, 49, 99, E2, B2, E6, 34, 4C, CF, 2A, 45, 87, 50, 57, D8, D7, 91, 8F, 9F, A0, 08, 9B, EF, 16, 0E, F9, 23, 33, FC, 19, 5B, C6, F6, 93, 5E, 1E, E0, 67, 20, 1D, 6E, 24, 2C, 0F, DE, 39, BF, F1, 1B, 07, 4A, C1, B0, D4, 6D, D2, 1F, 22, FD, C4, 04, AD, B5, 72, D1, 88, 56, 92, BC, 9A, C9, 01, A8, 0A, 2B, 46, 55, ED, C5, 61, 75, 2F, 90, F8, 82, 71, 74, 11, 83,

1C, A3, 43, DB, 06, 8D, 47, 81, 12, 52, AF, 02, 4D, E3, C2, 09, 0B, B7, B4, 17, 6A, C0, AE, A9, C3, C8, 48, 80, 4B, F5, F2, 8A, 8E, 37, E9, CB, 3F, 27, 65, 6B, 9E, CE, 18, 0D, BE, 36, 64, 94, DC, 8B, A5, 73, 58, 98, D6, 41, 03, D5, 4E, 42,
NL = 104
DU = 10
AI = 3

Appendix C

Starting simulated annealing . . .
Parameters:
Thread count: 8
Max outer loops in thread: 10,000
Max inner loops in thread: 100
Initial temperature: 100,000
Alpha parameter: 0.1
Max frozen loops: 1,000,000
cost = 5.88431×10^7 NL = 96 temperature = 100,000 Iteration = 8
cost = 5.34364×10^7 NL = 96 temperature = 100,000 Iteration = 11
. . .
cost = 1.17514×10^7 NL = 98 temperature = 100,000 Iteration = 262
cost = 1.15671×10^7 NL = 98 temperature = 100,000 Iteration = 269
cost = 1.21405×10^7 NL = 100 temperature = 100,000 Iteration = 277
cost = 1.24641×10^7 NL = 98 temperature = 100,000 Iteration = 280
cost = 1.13951×10^7 NL = 98 temperature = 100,000 Iteration = 282
cost = 1.13746×10^7 NL = 98 temperature = 100,000 Iteration = 283
cost = 1.30744×10^7 NL = 100 temperature = 100,000 Iteration = 285
cost = 1.23781×10^7 NL = 100 temperature = 100,000 Iteration = 286
cost = 1.10838×10^7 NL = 98 temperature = 100,000 Iteration = 289
cost = 1.1776×10^7 NL = 100 temperature = 100,000 Iteration = 292
cost = 1.07602×10^7 NL = 100 temperature = 100,000 Iteration = 296
cost = 1.08585×10^7 NL = 100 temperature = 100,000 Iteration = 298
cost = 1.05759×10^7 NL = 100 temperature = 100,000 Iteration = 334
cost = 1.08298×10^7 NL = 100 temperature = 100,000 Iteration = 349
cost = 1.06168×10^7 NL = 100 temperature = 100,000 Iteration = 355
cost = 1.05513×10^7 NL = 100 temperature = 100,000 Iteration = 367
cost = 1.05718×10^7 NL = 100 temperature = 100,000 Iteration = 368
cost = 1.0412×10^7 NL = 100 temperature = 100,000 Iteration = 370
. . .
cost = 6.95501×10^6 NL = 100 temperature = 10,000 Iteration = 759
cost = 6.97958×10^6 NL = 100 temperature = 10,000 Iteration = 797
cost = 6.92634×10^6 NL = 100 temperature = 10,000 Iteration = 800
. . .
cost = 2.60096×10^6 NL = 102 temperature = 1×10^{-93} Iteration = 57,822
cost = 2.60096×10^6 NL = 102 temperature = 1×10^{-93} Iteration = 58,090
SEARCH COST: 58,090
target sbox:
E1, 10, 76, 86, 92, 96, A9, 57, 7B, D9, 87, 91, 3D, C7, 06, 7D, DE, 49, 55, 80, F0, 25, 9E, 6B, 93, 64, BD, C1, 47, B4, 4D, 56, 07, 2F, 84, 23, B9, 21, B0, DC, 2D, 04, 0B, AA, 24, B5, 37, CF, 52, B6, 17, 34, 70, 08, AD, 48, D6, D5, CE, 09, 5C, F3, 0E, D2, D8, F8, 5B, B3, ED, EE, A8, B1, 5F, 95, 9D, 77, 00, BE, E4, 18, 61, DF, F1, CC, 9F, 5E, 6D, 2A, 20, 53, 7A, E2, 05, 1B, 42, 75, 3B, 4A, DD, 99, 7C, 2C, 46, 4F, FF, 35, 38, 8E, CA, D7, 7F, AE, 33, 58, 90, E0, FC, 2B, 3A, 67, 22, 02, 81, BB, 29, 1C, F5, 5D, D4, 36, CB, AC, DB, 45, C6, 2E, 14, 63, 1D, 9C, E8, B7, 94, EB, 54, F4, 8D, 01, 89, 31, FB, C8, 39, EC, AF, 69, E6, A6, A4, 43, 1E, 65, F2, 4E, 28, 8A, C9, F9, E7, 3C, 51, EA,

E5, 3F, 82, 5A, FE, F6, 97, F7, 68, 30, C5, 16, BA, C3, 26, A2, DA, 6F, A3, C4, 4C, BC, 40, 72, 11, 8B, 74, 1A, B2, AB, 0D, C0, 03, 3E, 19, A7, A0, 78, 12, EF, 4B, 85, 66, 27, 0F, 62, 8F, D3, FD, C2, 8C, D0, 79, 98, 0C, 1F, 50, 60, 9A, E9, 15, 9B, 41, 6C, 88, 0A, 73, A1, 6A, 32, 59, 6E, 71, E3, 83, FA, A5, BF, B8, 13, 7E, D1, CD, 44,
NL = 104
DU = 10
AI = 3

References

1. Menezes, A.J.; van Oorschot, P.C.; Vanstone, S.A.; van Oorschot, P.C.; Vanstone, S.A. *Handbook of Applied Cryptography*; CRC Press: Boca Raton, FL, USA, 2018; ISBN 978-0-429-46633-5.
2. Schneier, B. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*; Wiley: New York, NY, USA, 1996; ISBN 978-0-471-12845-8.
3. Kuznetsov, A.A.; Potii, O.V.; Poluyanenko, N.A.; Gorbenko, Y.I.; Kryvinska, N. *Stream Ciphers in Modern Real-Time IT Systems*; Studies in Systems, Decision and Control; Springer Nature: Cham, Switzerland, 2022; ISBN 978-3-030-79770-6.
4. Carlet, C.; Ding, C. Nonlinearities of S-Boxes. *Finite Fields Appl.* **2007**, *13*, 121–135. [CrossRef]
5. Carlet, C. Vectorial Boolean Functions for Cryptography. In *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*; Cambrige University Press: New York, NY, USA, 2006.
6. Matsui, M. Linear Cryptanalysis Method for DES Cipher. In *Advances in Cryptology, Proceedings of the EUROCRYPT '93: Workshop on the Theory and Application of Cryptographic Techniques Lofthus, Norway, 23–27 May 1993*; Helleseth, T., Ed.; Springer: Berlin/Heidelberg, Germany, 1994; pp. 386–397.
7. Mihailescu, M.I.; Nita, S.L. Linear and Differential Cryptanalysis. In *Pro Cryptography and Cryptanalysis: Creating Advanced Algorithms with C# and .NET*; Mihailescu, M.I., Nita, S.L., Eds.; Apress: Berkeley, CA, USA, 2021; pp. 457–481. ISBN 978-1-4842-6367-9.
8. Biham, E.; Perle, S. Conditional Linear Cryptanalysis—Cryptanalysis of DES with Less Than 242 Complexity. *IACR Trans. Symmetric Cryptol.* **2018**, *3*, 215–264. [CrossRef]
9. Freyre Echevarría, A. Evolución Híbrida de S-Cajas No Lineales Resistentes a Ataques de Potencia. Master's Thesis, Universidad de La Habana, Havana, Cuba, 2020.
10. Álvarez-Cubero, J. Vector Boolean Functions: Applications in Symmetric Cryptography. Ph.D. Thesis, Universidad Politécnica de Madrid, Madrid, Spain, 2015.
11. Picek, S.; Cupic, M.; Rotim, L. A New Cost Function for Evolution of S-Boxes. *Evol. Comput.* **2016**, *24*, 695–718. [CrossRef] [PubMed]
12. Freyre-Echevarría, A.; Martínez-Díaz, I.; Pérez, C.M.L.; Sosa-Gómez, G.; Rojas, O. Evolving Nonlinear S-Boxes with Improved Theoretical Resilience to Power Attacks. *IEEE Access* **2020**, *8*, 202728–202737. [CrossRef]
13. Ars, G.; Faugère, J.-C. *Algebraic Immunities of Functions over Finite Fields*; INRIA: Paris, France, 2005; p. 17.
14. Courtois, N.T.; Bard, G.V. Algebraic Cryptanalysis of the Data Encryption Standard. In *Cryptography and Coding, Proceedings of the 11th IMA International Conference, Cirencester, UK, 18–20 December 2007*; Galbraith, S.D., Ed.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 152–169.
15. Bard, G.V. *Algebraic Cryptanalysis*; Springer: Boston, MA, USA, 2009; ISBN 978-0-387-88756-2.
16. Daemen, J.; Rijmen, V. Specification of Rijndael. In *The Design of Rijndael: The Advanced Encryption Standard (AES)*; Daemen, J., Rijmen, V., Eds.; Information Security and Cryptography; Springer: Berlin/Heidelberg, Germany, 2020; pp. 31–51. ISBN 978-3-662-60769-5.
17. Courtois, N.T.; Pieprzyk, J. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In *Advances in Cryptology, Proceedings of the ASIACRYPT 2002: 8th International Conference on the Theory and Application of Cryptology and Information Security Queenstown, New Zealand, 1–5 December 2002*; Zheng, Y., Ed.; Springer: Berlin/Heidelberg, Germany, 2002; pp. 267–287.
18. Gorbenko, I.; Kuznetsov, A.; Gorbenko, Y.; Pushkar'ov, A.; Kotukh, Y.; Kuznetsova, K. Random S-Boxes Generation Methods for Symmetric Cryptography. In Proceedings of the 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON), Lviv, Ukraine, 2–6 July 2019; pp. 947–950.
19. Clark, A.J. Optimisation Heuristics for Cryptology. Ph.D. Thesis, Queensland University of Technology, Brisbane City, Australia, 1998.
20. Millan, W. How to Improve the Nonlinearity of Bijective S-Boxes. In *Information Security and Privacy, Proceedings of the Third Australasian Conference, ACISP'98, Brisbane, Australia, 13–15 July 1998*; Boyd, C., Dawson, E., Eds.; Springer: Berlin/Heidelberg, Germany, 1998; pp. 181–192.
21. Clark, J.A.; Jacob, J.L.; Stepney, S. The Design of S-Boxes by Simulated Annealing. In Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753), Portland, OR, USA, 19–23 June 2004; Volume 2, pp. 1533–1537.
22. Burnett, L.D. Heuristic Optimization of Boolean Functions and Substitution Boxes for Cryptography. Ph.D. Thesis, Queensland University of Technology, Brisbane City, Australia, 2005.
23. Delahaye, D.; Chaimatanan, S.; Mongeau, M. Simulated Annealing: From Basics to Applications. In *Handbook of Metaheuristics*; [Gendreau, M., Potvin, J.-Y., Eds.; International Series in Operations Research & Management Science; Springer International Publishing: Cham, Switzerland, 2019; Volume 272, pp. 1–35. ISBN 978-3-319-91085-7.

24. McLaughlin, J.; Clark, J.A. Using Evolutionary Computation to Create Vectorial Boolean Functions with Low Differential Uniformity and High Nonlinearity. *arXiv* **2013**, arXiv:1301.6972.
25. Kuznetsov, A.; Wieclaw, L.; Poluyanenko, N.; Hamera, L.; Kandiy, S.; Lohachova, Y. Optimization of a Simulated Annealing Algorithm for S-Boxes Generating. *Sensors* **2022**, *22*, 6073. [CrossRef] [PubMed]
26. Freyre Echevarría, A.; Martínez Díaz, I. A New Cost Function to Improve Nonlinearity of Bijective S-Boxes. *Symmetry* **2020**, *12*, 1896. [CrossRef]
27. Kuznetsov, A.; Poluyanenko, N.; Kandii, S.; Zaichenko, Y.; Prokopovich-Tkachenko, D.; Katkova, T. Optimizing the Local Search Algorithm for Generating S-Boxes. In Proceedings of the 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S T), Kharkiv, Ukraine, 5–7 October 2021; pp. 458–464.
28. Freyre-Echevarría, A.; Alanezi, A.; Martínez-Díaz, I.; Ahmad, M.; Abd El-Latif, A.A.; Kolivand, H.; Razaq, A. An External Parameter Independent Novel Cost Function for Evolving Bijective Substitution-Boxes. *Symmetry* **2020**, *12*, 1896. [CrossRef]
29. Millan, W.; Clark, A.; Dawson, E. Boolean Function Design Using Hill Climbing Methods. In *Information Security and Privacy, Proceedings of the 4th Australasian Conference, ACISP'99 Wollongong, NSW, Australia, 7–9 April 1999*; Pieprzyk, J., Safavi-Naini, R., Seberry, J., Eds.; Springer: Berlin/Heidelberg, Germany, 1999; pp. 1–11.
30. Millan, W.; Clark, A. Smart Hill Climbing Finds Better Boolean Functions. In *Workshop on Selected Areas in Cryptology*; Queensland University of Technology: Queensland, Australia, 1997.
31. Souravlias, D.; Parsopoulos, K.E.; Meletiou, G.C. Designing Bijective S-Boxes Using Algorithm Portfolios with Limited Time Budgets. *Appl. Soft Comput.* **2017**, *59*, 475–486. [CrossRef]
32. Wang, J.; Zhu, Y.; Zhou, C.; Qi, Z. Construction Method and Performance Analysis of Chaotic S-Box Based on a Memorable Simulated Annealing Algorithm. *Symmetry* **2020**, *12*, 2115. [CrossRef]
33. Friedli, S.; Velenik, Y. *Statistical Mechanics of Lattice Systems: A Concrete Mathematical Introduction*, 1st ed.; Cambridge University Press: Cambridge, UK, 2017; ISBN 978-1-107-18482-4.
34. Eremia, M.; Liu, C.-C.; Edris, A.-A. Heuristic Optimization Techniques. In *Advanced Solutions in Power Systems: HVDC, FACTS, and Artificial Intelligence*; IEEE: Manhattan, NY, USA, 2016; pp. 931–984. ISBN 978-1-119-17533-9.
35. Laskari, E.C.; Meletiou, G.C.; Vrahatis, M.N. Utilizing Evolutionary Computation Methods for the Design of S-Boxes. In Proceedings of the 2006 International Conference on Computational Intelligence and Security, Guangzhou, China, 3–6 November 2006; Volume 2, pp. 1299–1302.
36. Tesar, P. A New Method for Generating High Non-Linearity S-Boxes. *Radioengineering* **2010**, *19*, 23–26.
37. Eiben, A.E.; Smith, J.E. Genetic Algorithms. In *Introduction to Evolutionary Computing*; Eiben, A.E., Smith, J.E., Eds.; Natural Computing Series; Springer: Berlin/Heidelberg, Germany, 2003; pp. 37–69. ISBN 978-3-662-05094-1.
38. Ivanov, G.; Nikolov, N.; Nikova, S. Cryptographically Strong S-Boxes Generated by Modified Immune Algorithm. In *Cryptography and Information Security in the Balkans, Proceedings of the Second International Conference, BalkanCryptSec 2015, Koper, Slovenia, 3-4 September 2015*; Pasalic, E., Knudsen, L.R., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 31–42.