sensors

Article

Optimization of a Simulated Annealing Algorithm for S-Boxes Generating

Alexandr Kuznetsov [1,2,3], Lukasz Wieclaw [4,*], Nikolay Poluyanenko [2,3], Lukasz Hamera [4], Sergey Kandiy [2,3] and Yelyzaveta Lohachova [2]

1 Department of Political Sciences, Communication and International Relations, University of Macerata, Via Crescimbeni, 30/32, 62100 Macerata, Italy
2 Department of Information and Communication Systems Security, Faculty of Comupter Science, V. N. Karazin Kharkiv National University, 4 Svobody Sq., 61022 Kharkiv, Ukraine
3 Department of Information Systems and Technologies Security, JSC "Institute of Information Technologies", Bakulin St., 12, 61166 Kharkiv, Ukraine
4 Department of Computer Science and Automatics, Faculty of Mechanical Engineering and Computer Science, University of Bielsko-Biala, Willowa 2, 43-300 Bielsko-Biala, Poland
* Correspondence: lwieclaw@ath.edu.pl

Abstract: Cryptographic algorithms are used to ensure confidentiality, integrity and authenticity of data in information systems. One of the important areas of modern cryptography is that of symmetric key ciphers. They convert the input plaintext into ciphertext, representing it as a random sequence of characters. S-boxes are designed to complicate the input–output relationship of the cipher. In other words, S-boxes introduce nonlinearity into the encryption process, complicating the use of different methods of cryptanalysis (linear, differential, statistical, correlation, etc.). In addition, S-boxes must be random. This property means that nonlinear substitution cannot be represented as simple algebraic constructions. Random S-boxes are designed to protect against algebraic methods of cryptanalysis. Thus, generation of random S-boxes is an important area of research directly related to the design of modern cryptographically strong symmetric ciphers. This problem has been solved in many related works, including some using the simulated annealing (SA) algorithm. Some works managed to generate 8-bit bijective S-boxes with a nonlinearity index of 104. However, this required enormous computational resources. This paper presents the results of our optimization of SA via various parameters. We were able to significantly reduce the computational complexity of substitution generation with SA. In addition, we also significantly increased the probability of generating the target S-boxes with a nonlinearity score of 104.

Keywords: simulated annealing algorithm; nonlinear substitutions; iterative search; computational complexity; S-box

check for updates

Citation: 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. https://doi.org/10.3390/s22166073

Academic Editor: Wei Yi

Received: 15 July 2022
Accepted: 11 August 2022
Published: 14 August 2022

1. Introduction

Data encryption algorithms with symmetric keys are used in modern computer systems to ensure confidentiality, integrity, authenticity and other information security services [1–3]. The main condition for using such encryption algorithms is the availability of a secret key, which is identical for the sender and receiver of data.

An important component of most modern secret-key ciphers is nonlinear substitution (S-boxes). These boxes are designed to introduce complex nonlinear relationships into the plaintext–ciphertext relationship. In fact, S-boxes play a crucial role in providing cryptographic strength. Using classical terms of the theory of secret systems [4], S-boxes provide the confusion property, which plays a crucial role in protecting against differential, linear, statistical, correlation and many other types of cryptanalysis [1,5].

According to modern concepts, nonlinear substitutions in cryptography should be random [2]. Nonrandom methods of substitution generation can cause vulnerabilities in cryptoalgorithms [6,7]. For example, the well-known encryption algorithm AES [8] uses algebraic (not random) methods of S-box generation, and this fact was the reason for the appearance of algebraic cryptanalysis [9,10]. The simplicity of algebraic construction of S-box cipher AES is used to criticize this cryptoalgorithm. Thus, methods of generating S-boxes should be based on the use of random substitution processes.

One of the well-known methods for generating S-boxes is the simulated annealing (SA) algorithm [11–14]. The name of this algorithm comes from annealing in metallurgy, when heating and controlled cooling of a metal determine its physical properties [15]. These processes are simulated by a computer program for nonlinear substitution generation. The initial temperature sets the probability of random change in the S-box. This temperature gradually decreases, which leads to a gradual reduction in random changes. As a result, the process solidifies, and we have a final stand, the cryptographic properties of which are determined by the SA parameters.

It should be noted that SA suffers from the capture of local optima [15,16]. Nevertheless, this algorithm is well suited for solving the substitution generation problem. The global optimum in this problem corresponds to the maximum nonlinearity of the substitution, and this is achieved by using special algebraic structures in the finite field. For example, known algebraic constructs from [17,18] provide maximum nonlinearity of Boolean mappings (these very constructs were used to generate AES cipher substitutions [8]). Such mathematical constructs are described by simple algebraic equations, which can potentially be used to find cipher vulnerabilities [6,7,9]. We are interested in generating random highly nonlinear S-boxes that have no hidden mathematical structures, i.e., we are looking for local optimums of nonlinearity, and SA is well suited for this problem.

Substitution generation using SA has been investigated by many authors. However, the computational complexity of the known solutions is very high. For example, it took more than 3 million iterations to generate an 8-bit bijective substitution with nonlinearity 104 in [12]. In addition, the probability of generating a target S-box is also very low. For example, in [11,19], the probability of generating a statement with nonlinearity 102 was about 0.5%.

In this paper, we optimized the SA parameters to generate target S-boxes (hereinafter, the target refers to an 8-bit bijective substitution with nonlinearity 104). Our optimization of SA allowed us to significantly reduce the computational complexity (about 450 thousand iterations are required) and increase the probability (more than 50%) of generating the target S-boxes.

2. Related Work

Evolutionary techniques of computational intelligence are used to solve complex computational problems related to mathematical optimization and search for the best element by some criterion from some set of available alternatives [20,21]. Evolutionary algorithms are used to solve various combinatorial optimization problems [22,23]. For example, these include global and engineering optimization problems [24], production re-planning in Industry 4.0 [25], optimization [26] and routing problems [27], and industrial production optimization [28].

One of the most efficient methods for solving global optimization problems (especially discrete and combinatorial optimization) is SA. This algorithm is inspired by the natural processes that occur in the annealing of metals. The algorithm is based on the simulation of the physical process that occurs when a substance crystallizes. It is assumed that the atoms of matter are almost lined up in a crystal lattice, but transitions of individual atoms from one cell to another are still allowed. The higher the temperature, the greater the activity of the atoms. The temperature is gradually lowered, which leads to a decrease in the probability of transitions. A stable crystal lattice corresponds to the minimum energy of the atoms. In computational intelligence, this process is simulated as a computational algorithm for solving a global optimization problem, i.e., it is necessary to find the point (set of points) where the minimum (or maximum) of some target function is reached.

The first works that used SA for the problem of generating nonlinear S-boxes were the works of John A. Clark [11,19]. The author managed to generate an 8-bit substitution with a nonlinearity of 102. In addition, he proposed a cost function for SA, which was used in further related works [29–32]. In [30,32,33], other forms of the cost function were investigated. In [12–14,34], SA for generation of highly nonlinear S-boxes was investigated. However, the computational complexity of solving this problem turned out to be very high. For example, in [12], they managed to generate an 8-bit bijective S-box with nonlinearity 104, but it required more than 3 million iterations. In [13], the authors managed to generate a substitution with nonlinearity 100, which is significantly lower than other known results. SA was also used in [14] to generate permutations, but only nonlinearity 92 was achieved.

Thus, SA is used to generate nonlinear substitutions in cryptography. However, the computational complexity of the generation algorithm is very high. In addition, the probability of generating a target S-box is very low. For example, in [11,19], the probability of generating a statement with nonlinearity 102 was about 0.5%.

3. Materials and Methods

The main characteristic of heuristic search is the cost function $C(S)$, which displays the state of the system in some natural way.

We used the function from [33,35] as the substitution cost function:

$$C(S) = \sum_{i=1}^{255} ||\max(WHT)| - X|^{R}. \tag{1}$$

where:

- WHT—Walsh–Hadamard spectral coefficients;
- X and R—some parameters of the target function WHS.
- As the optimal parameters of the function (1) selected [33,35]:
- $X = 36$ as the maximum permissible value, which reduces $C(S)$, but does not lead to a significant effect on its adequate relationship with the nonlinearity of the S-box;
- $R = 4$ as the maximum allowable value, increasing the range of function values $C(S)$, which can improve the "sensitivity" of S-box formation algorithms.

Note that when calculating the cost function $C(S)$, the nonlinearity of the S-box was simultaneously calculated:

$$N_f = \frac{1}{2} \cdot (2^n - \max(WHT)) = 128 - \frac{1}{2} \cdot \max(WHT). \tag{2}$$

The main advantage of SA is its ability to escape from local optima. This is achieved due to the ability of the algorithm to take some deteriorating steps in the local understanding but ensure that the algorithm advances and finds a better state.

The first application of the simulated annealing algorithm to the problem of S-box generation was given in [8]. At the beginning of the algorithm, the initial solution S_{best_sbox}, which provides, firstly, the property of bijectivity of the S-block and, secondly, its random nature, is formed. Then, a slight modification of the current state is performed. The new S-block state will be denoted as S_n.

After each modification, the cost Function (1) is calculated for S_n. This value is compared with the previous best solution, i.e., with the value of the cost function for S_{best_sbox}. If Condition (3) holds,

$$C(S_n) \leq C(S_{best\_sbox}), \tag{3}$$

then the algorithm takes $S_{best\_sbox} = S_n$. Using Condition (3) increases the number of possible solutions.

The main advantage of SA is the possibility of making a worsening decision, i.e., one that does not satisfy Condition (3). In our algorithm, if Condition (3) is not satisfied, the algorithm makes a worsening decision $S_{best\_sbox} = S_n$ with Probability (4):

$$\Pr(S_{best\_sbox} = S_n) = e^{\left(\frac{C(S_{best\_sbox})-C(S_n)}{T_i}\right)}, \tag{4}$$

where $T_i = \alpha \cdot T_{i-1}$ is the temperature equivalent in the process of metal annealing. In our case, this is a parameter characterizing the probability of deterioration of the current state.

The pseudocode of the implemented SA is shown in Figure 1.

Algorithm 1 SA algorithm

```
1:  $S_{best\_sbox} \leftarrow S_0$
2:  $S \leftarrow S_{best\_sbox}$
3:  $T \leftarrow T_0$
4:  $k^0_{int} \leftarrow 0$
5:  for $j = 1, k_{out}$ do
6:      for $i = 1, k_{int}$ do
7:          $S \leftarrow$ Modiffication($S_{best\_sbox}$)
8:          if $C(S) \leq C(S_{best\_sbox})$ or random(0...1)<Pr($S, S_{best\_sbox}, T$) then
9:              $S_{best\_sbox} \leftarrow S$
10:             if $N(S) \geq 104$ then return $S_{best\_sbox}$
11:             $k^0_{int} = 0$
12:         else
13:             $k^0_{int} = k^0_{int} + 1$
14:             if $k^0_{int} \geq k_{froz}$ then return 0
15:     $T = \alpha \cdot T$
16: return 0
```

Figure 1. Pseudocode of the proposed annealing simulation algorithm.

At each value of the current temperature T_i, the algorithm performs k_{int} iterations (let us call them inner cycles). The number of changes in the current temperature is determined by the parameter k_{out} (let us call it the number of external cycles). In order to limit the number of external iterations that do not yield improvements, we also introduced the parameter k_{froz}—the maximum number of external iterations without improvements.

The implemented algorithm of S-boxes generation was adapted to run in multi-threaded search mode.

4. Test Cases

When implementing the simulated annealing algorithm for S-box generation, we used the following initial parameters:

- K_{THREAD}—the number of threads in which the simultaneous search takes place. In our case, K_{THREAD} = 30, which corresponded to the maximum number of threads supported by the computer's processor;
- T_0—initial "temperature" value. It is stated in [36] that T_0 should provide an initial worst-case decision probability of 50–80%. We investigated the search efficiency at different values of T_0;
- α—"cooling coefficient", which determines how much the temperature decreases at each iteration of the algorithm. We investigated the search efficiency at different values of α;
- k_{int}—parameter, which specifies the number of internal cycles that the local search algorithm can perform at each temperature. We applied k_{int} = 650 (i.e., the total number of internal tests was . 30×650 = 19,500); Stopping criteria. The stopping criteria used were as follows:

- N_f—the target value of the nonlinearity (4) of the S-box. In our experiments, we limited ourselves to the value $N_f = 104$, i.e., the search stops when S_n with nonlinearity 104 is found;
- k_{out}—the maximum number of external cycles, i.e., how many times the SA algorithm was allowed to lower the temperature and continue searching before it stopped. We used k_{out} = 50;
- k_{froz}—the number of consecutive outer cycles in which no improvement of the cost function was found. We used k_{froz} = 5.

The individual parameters of the algorithm (k_{int}, k_{out}, k_{froz}) were chosen from the considerations given in [35].

The initial temperature varied from a value where the probability of making the worst decision was almost zero to a higher one where the probability was close to one.

The increase in T_0 was performed according to the rule:

$$T_0^{i+1} = 1.13 \cdot T_0^i. \quad (5)$$

For each T_0^i of (5), 100 runs of the simulated annealing algorithm were performed.

The parameter α varied from 0.6 to 0.95:

- for α =0.6, 10,100 runs of the search algorithm were performed;
- for α =0.7, 7600 launches were performed;
- for α =0.8, 5700 launches were performed;
- for α =0.9, 8200 launches were performed;
- for α =0.95, 8200 launches were performed.

The constraint k_{froz} = 5 resulted in the loss of some solutions for which the algorithm could still find the target S-box with nonlinearity 104. However, the expediency of further search was considered small compared to the time spent.

5. Results

The first part of the experiments consisted in estimating the number of runs of the search algorithm for which no improvement of the cost function was found for a long time. The obtained results are shown in Table 1.

Table 1. Distribution of the number of search algorithm runs for which no improvement of the cost function was found.

α	**The Number of External Loops of the Algorithm for Which No Improvement of the Cost Function Was Found**											
	0		**1**		**2**		**3**		**4**		**5**	
0.6	4281	42%	3042	30%	1509	15%	725	7%	402	4%	141	1.4%
0.7	2722	36%	2510	33%	1245	16%	641	8%	355	5%	127	1.7%
0.8	1362	24%	2060	36%	1244	22%	592	10%	322	6%	120	2.1%
0.9	2168	26%	1671	20%	1634	20%	1466	18%	955	12%	306	3.7%
0.95	2299	28%	1684	21%	1333	16%	1163	14%	1167	14%	554	6.8%

As we can see from the data in Table 1, there is an increase in time with increasing α in which no improvement in the cost function was found. This can be explained by an increase in the share of accepted value function deteriorations, which in turn leads to an increase in the probability of exiting the local minimum.

The second part of the experiments consisted in estimating the probability of forming the target S-box. The results are shown in Figures 2–5. The probabilities were estimated as the ratio of the number of generated target S-boxes to the total number of runs of the algorithm. Additionally, we measured the average time to generate substitutions. The results obtained are shown in Figures 6–9. The average generation time includes the time spent on unsuccessful runs of the search algorithm. For all graphs in Figures 2–9, we additionally present the trend line.

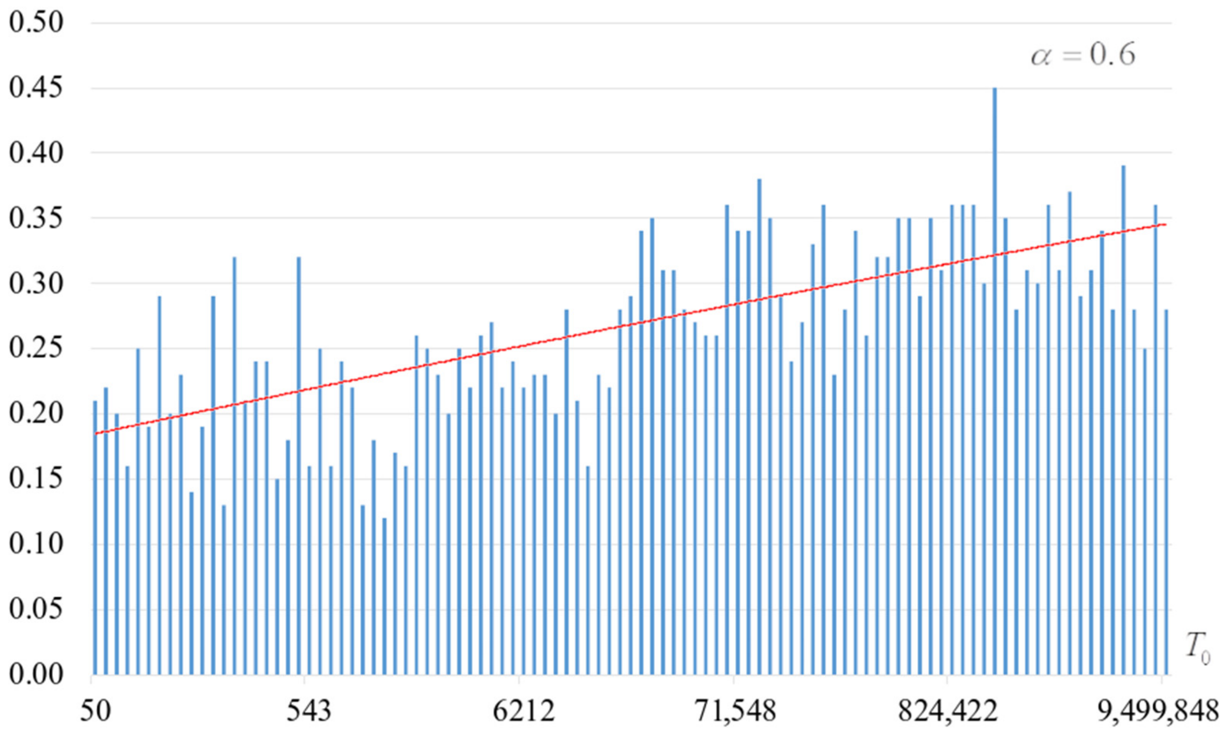

Figure 2. The probability of forming the target S-box at $\alpha = 0.6$.

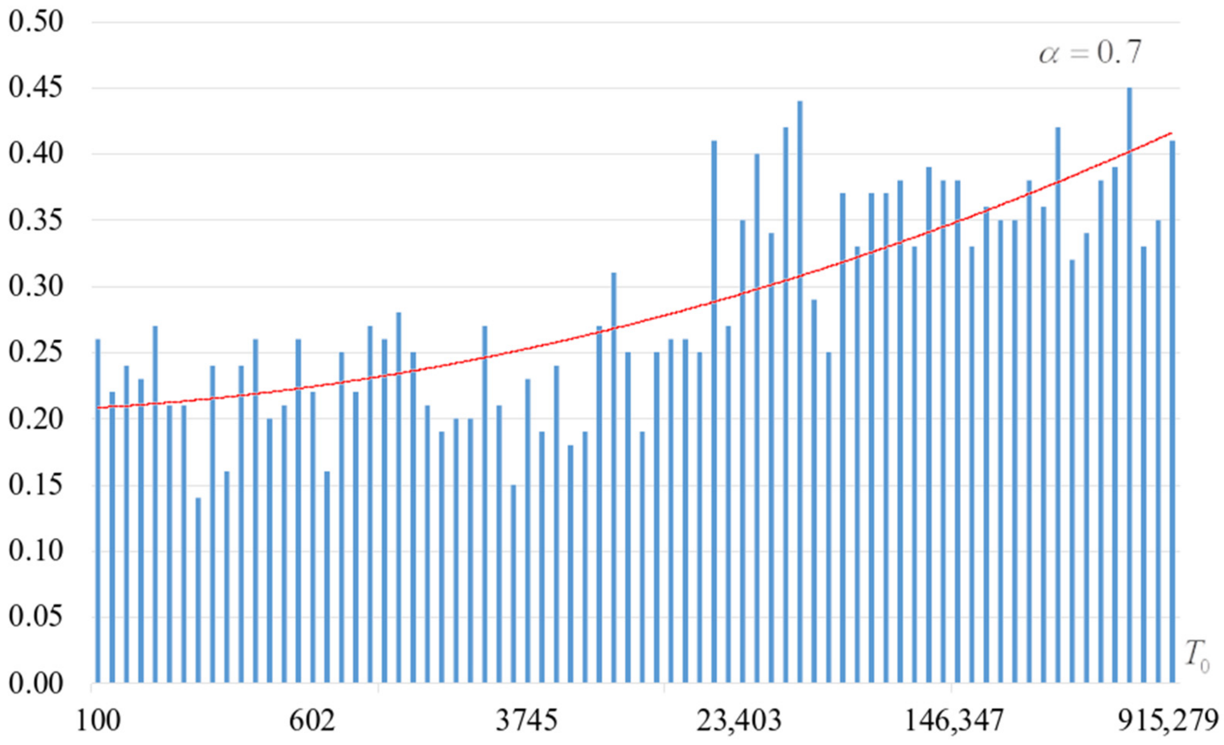

Figure 3. The probability of forming the target S-box at $\alpha = 0.7$.

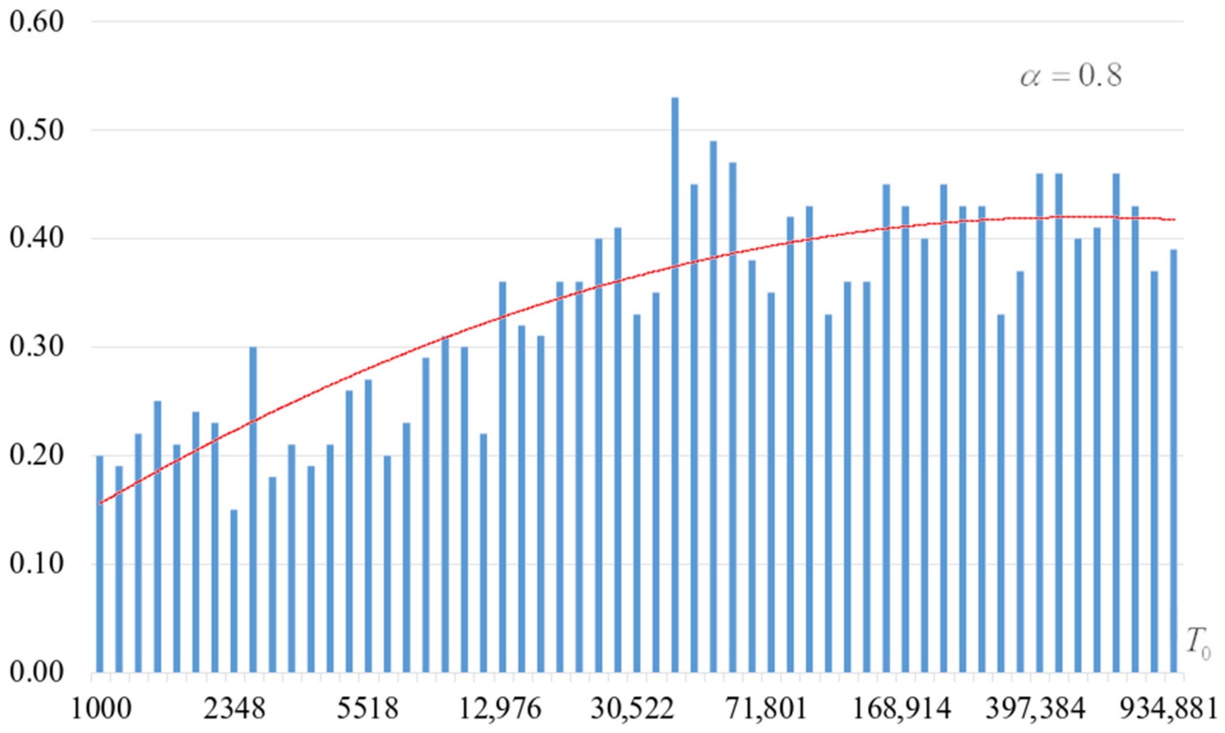

Figure 4. The probability of forming the target S-box at $\alpha = 0.8$.

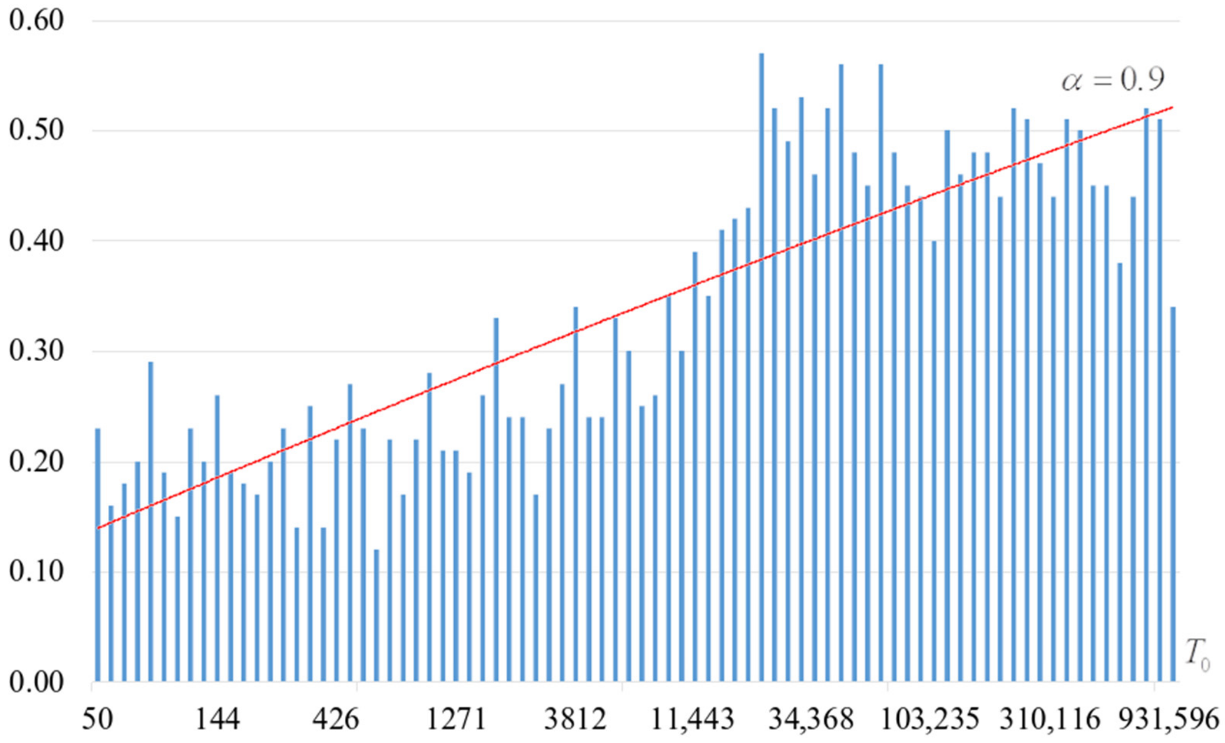

Figure 5. The probability of forming the target S-box at $\alpha = 0.9$.

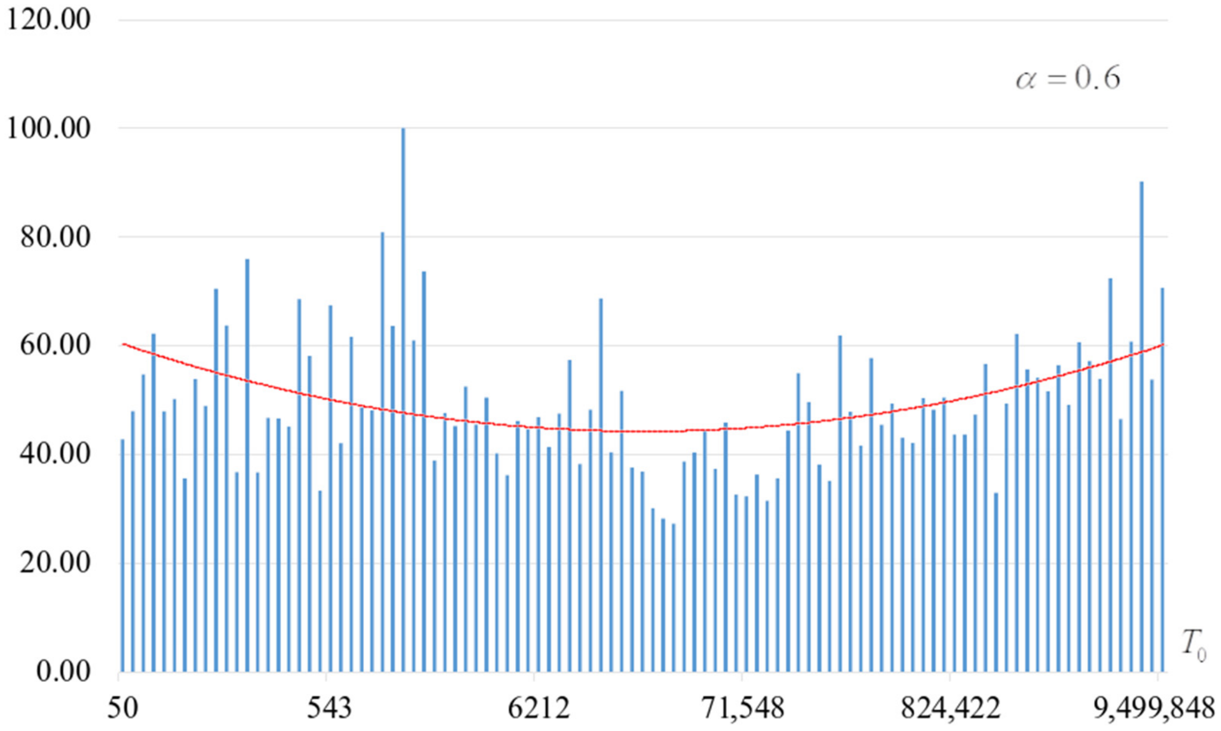

Figure 6. Average time (s) of target S-box formation at $\alpha = 0.6$.

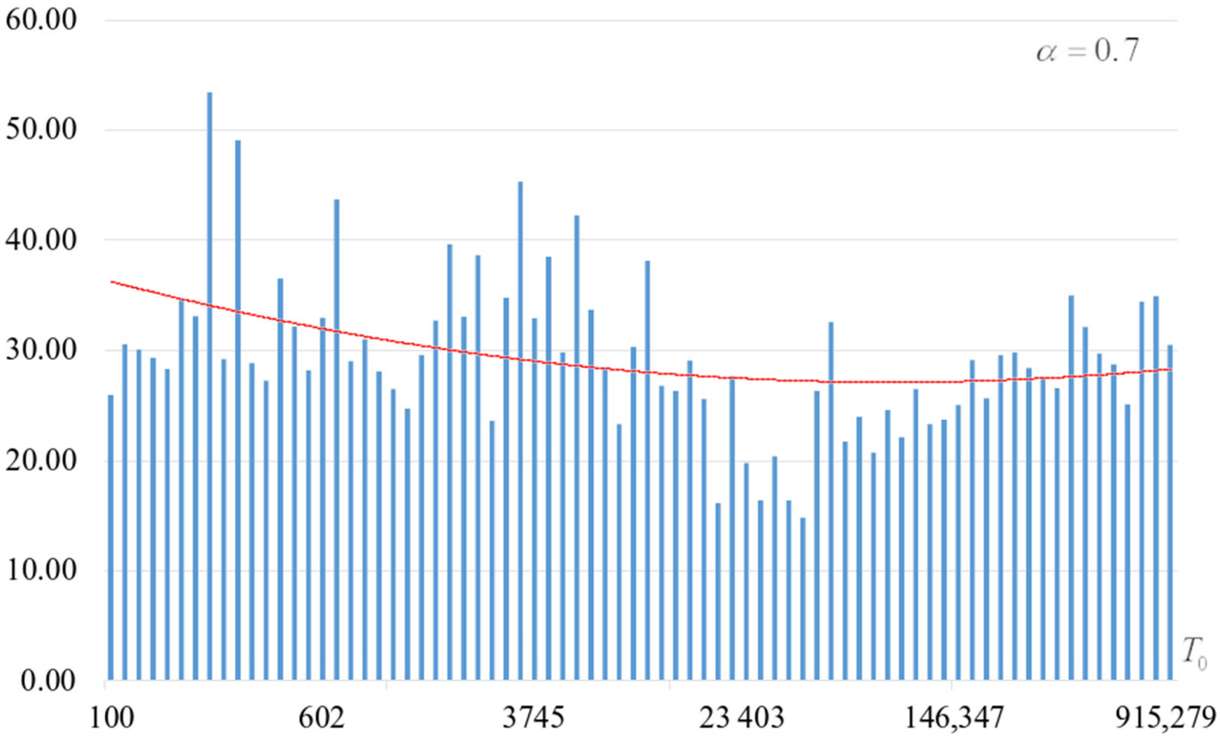

Figure 7. Average time (s) of target S-box formation at $\alpha = 0.7$.

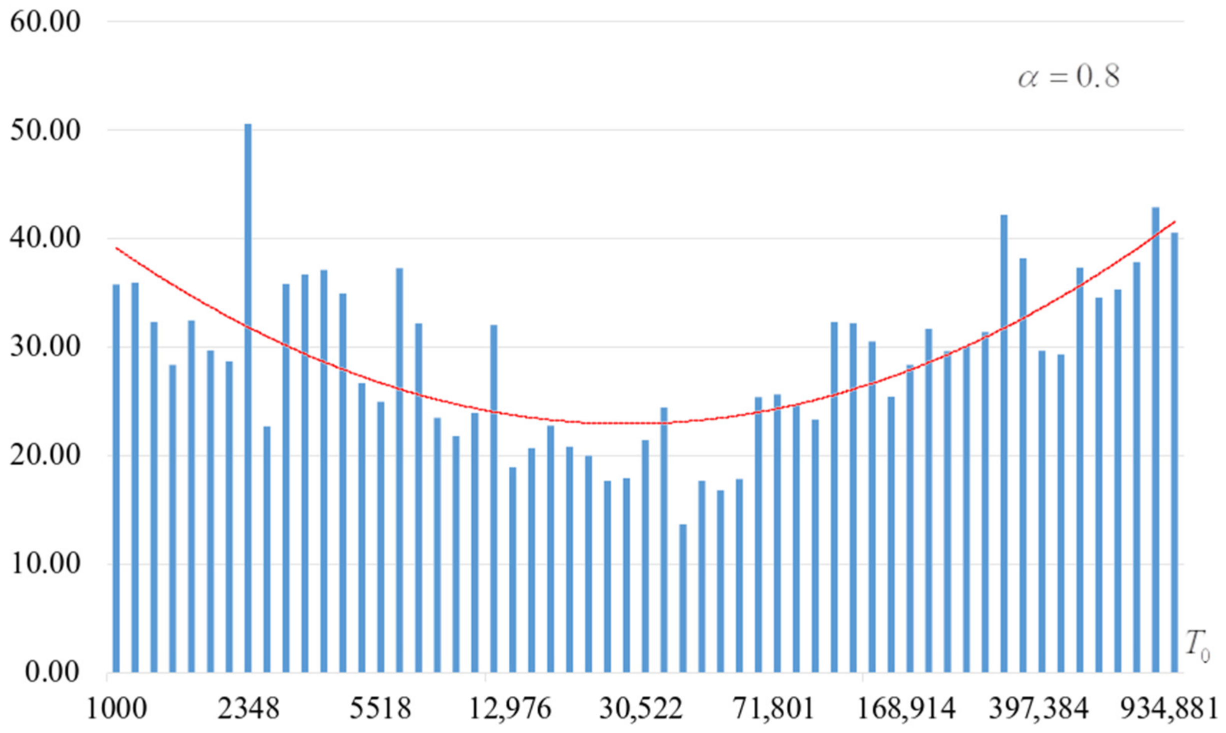

Figure 8. Average time (s) of target S-box formation at $\alpha = 0.8$.

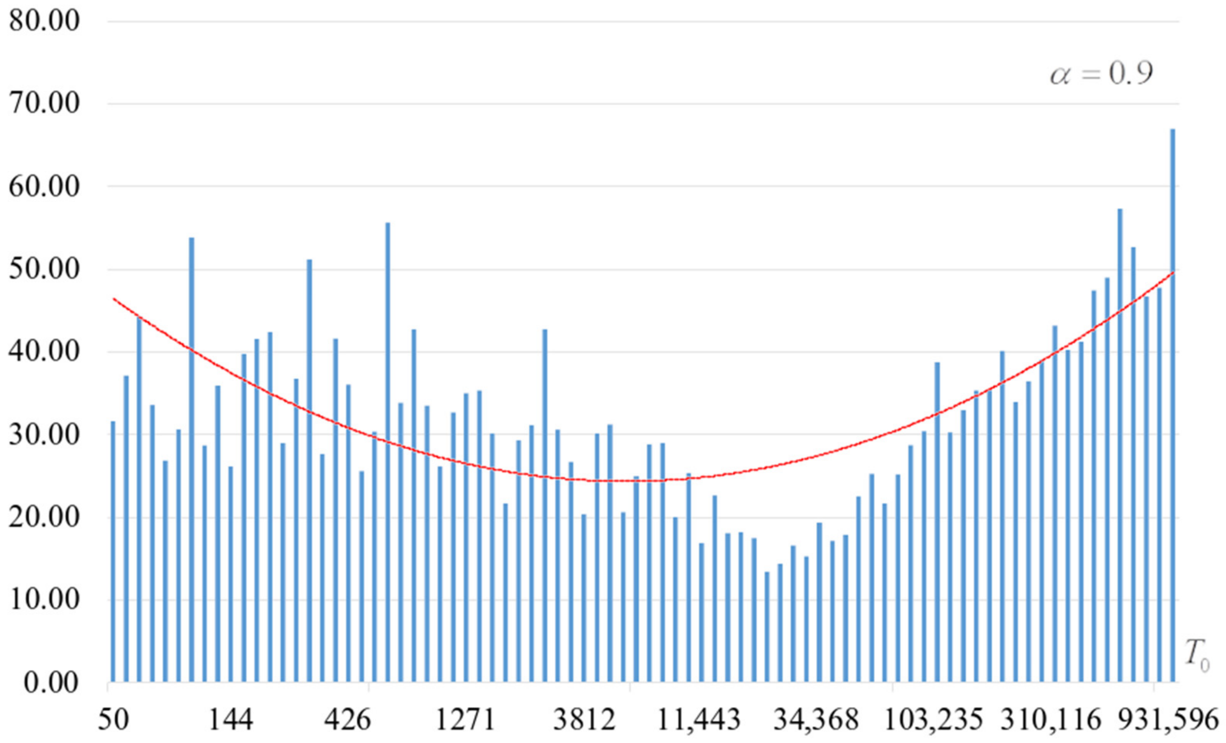

Figure 9. Average time (s) of target S-box formation at $\alpha = 0.9$.

To detail the obtained results, Figures 10–13 show the dependencies of the number of iterations of the outer loop (until one of the criteria for stopping the algorithm is fulfilled):

- The upper curve (red) corresponds to the maximum number of iterations;
- The middle curve (yellow) corresponds to the average number of iterations;
- The lower curve (green) corresponds to the minimum number of iterations.

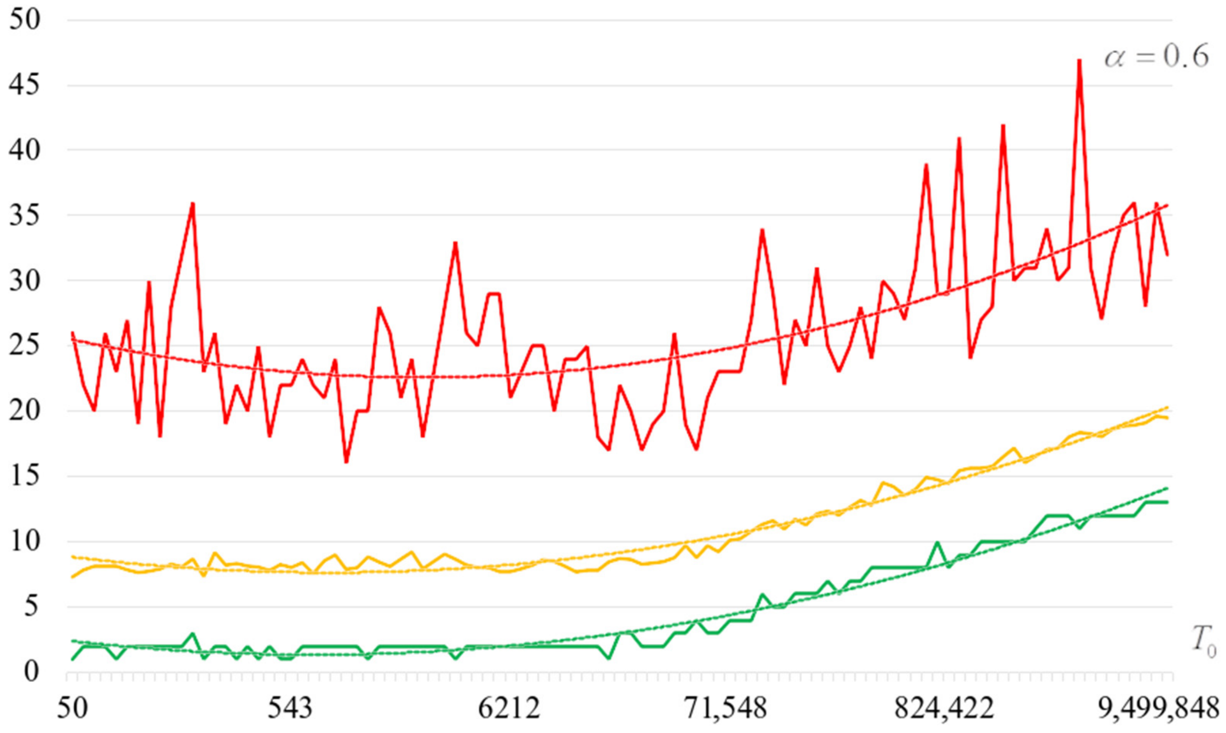

Figure 10. The dependence of the number of iterations of an external loop (until one of the criteria for stopping the algorithm is met), $\alpha = 0.6$.

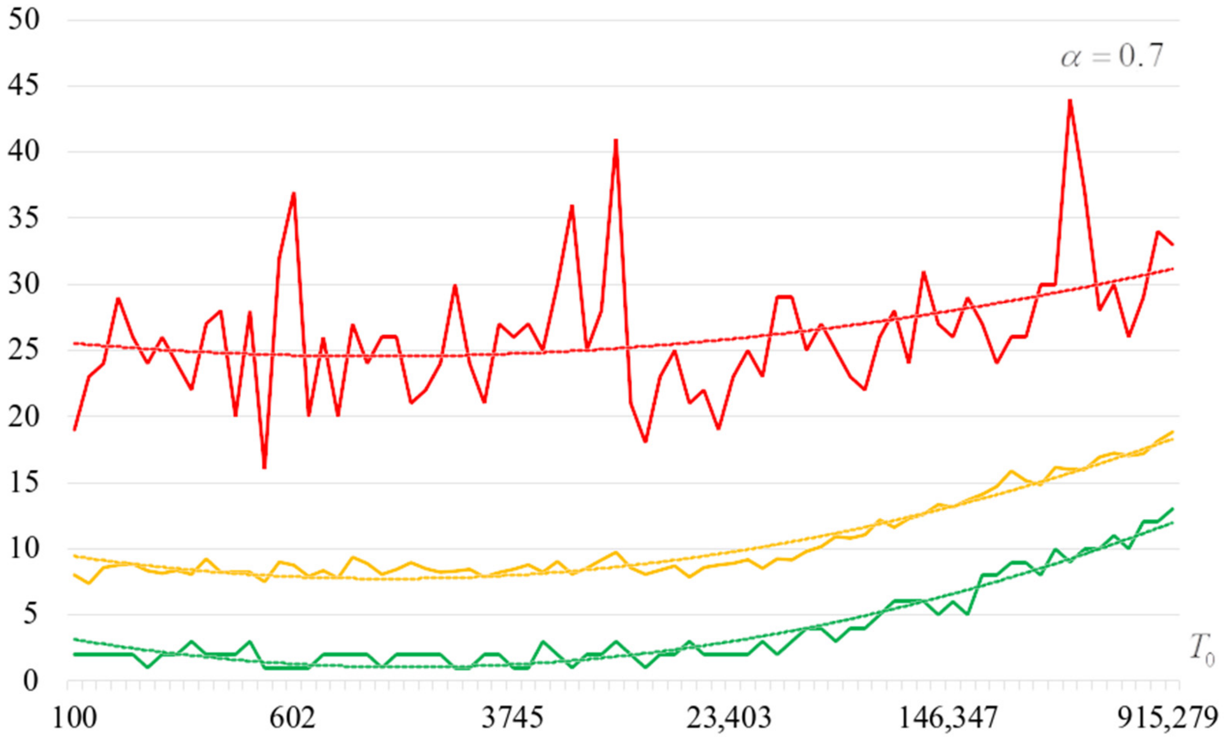

Figure 11. The dependence of the number of iterations of an external loop (until one of the criteria for stopping the algorithm is met), $\alpha = 0.7$.

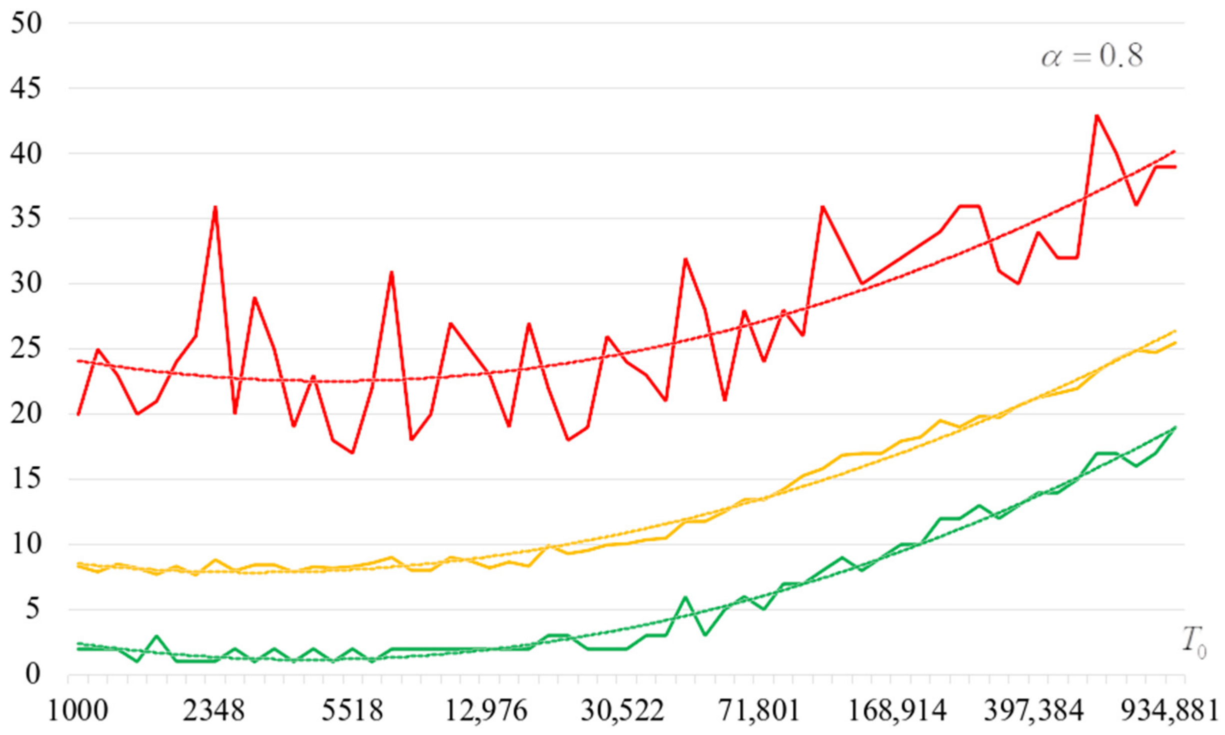

Figure 12. The dependence of the number of iterations of an external loop (until one of the criteria for stopping the algorithm is met), $\alpha = 0.8$.

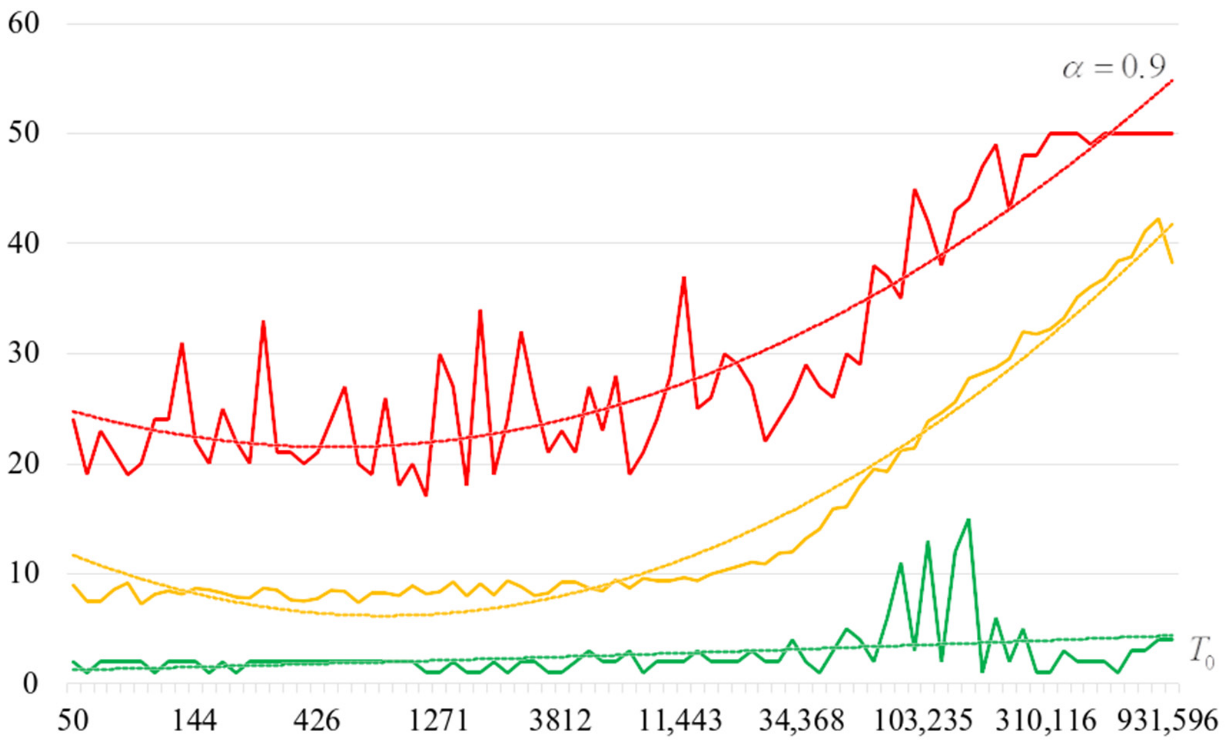

Figure 13. The dependence of the number of iterations of an external loop (until one of the criteria for stopping the algorithm is met), $\alpha = 0.9$.

The most interesting dependencies are shown in Figures 14–17. These dependencies are dependences of number of iterations of external loop (until one of the criteria of algorithm stopping is fulfilled) under the condition of successful start of the algorithm.

In other words, dependencies in Figures 14–17 correspond to the cases when running the search algorithm resulted in generation of the target S-box:

- The upper curve (red) corresponds to the maximum number of iterations;
- The middle curve (yellow) corresponds to the average number of iterations;
- The lower curve (green) corresponds to the minimum number of iterations.

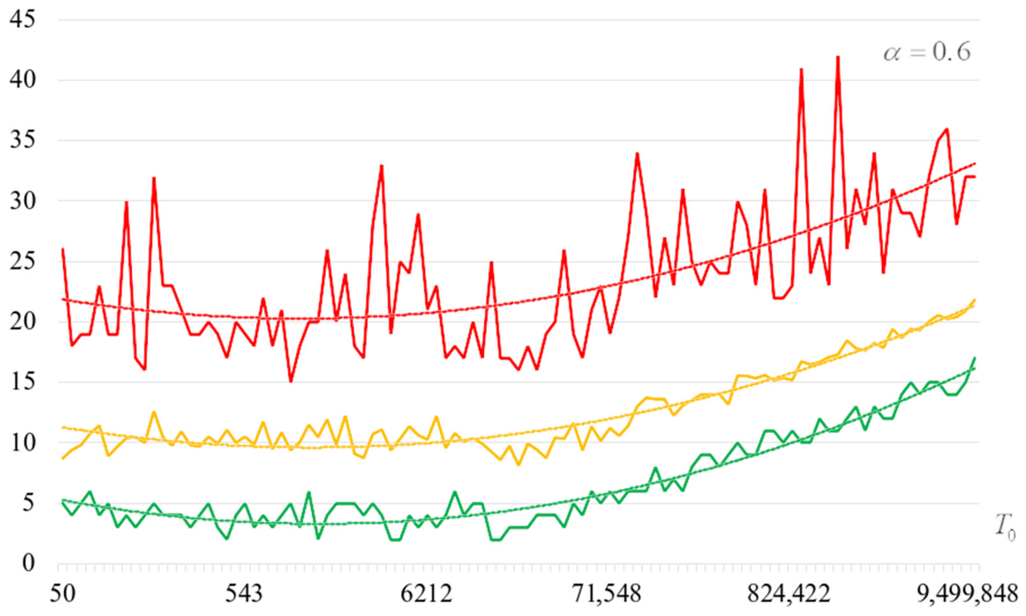

Figure 14. The dependencies of the number of iterations of the outer loop (until one of the criteria for stopping the algorithm is fulfilled) if the algorithm is successfully started, $\alpha = 0.6$.

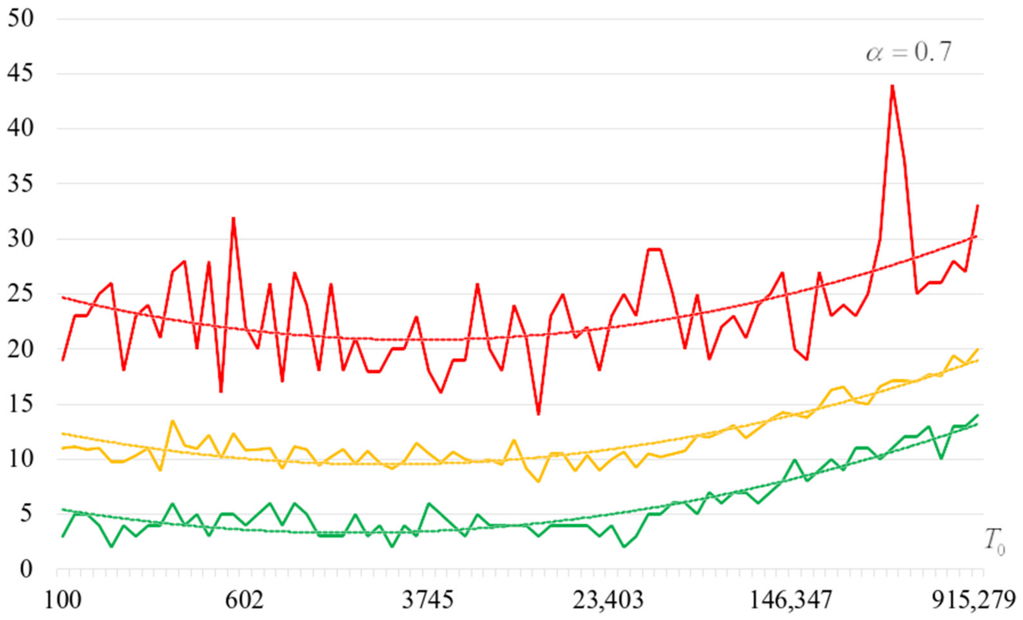

Figure 15. The dependencies of the number of iterations of the outer loop (until one of the criteria for stopping the algorithm is fulfilled) if the algorithm is successfully started, $\alpha = 0.7$.

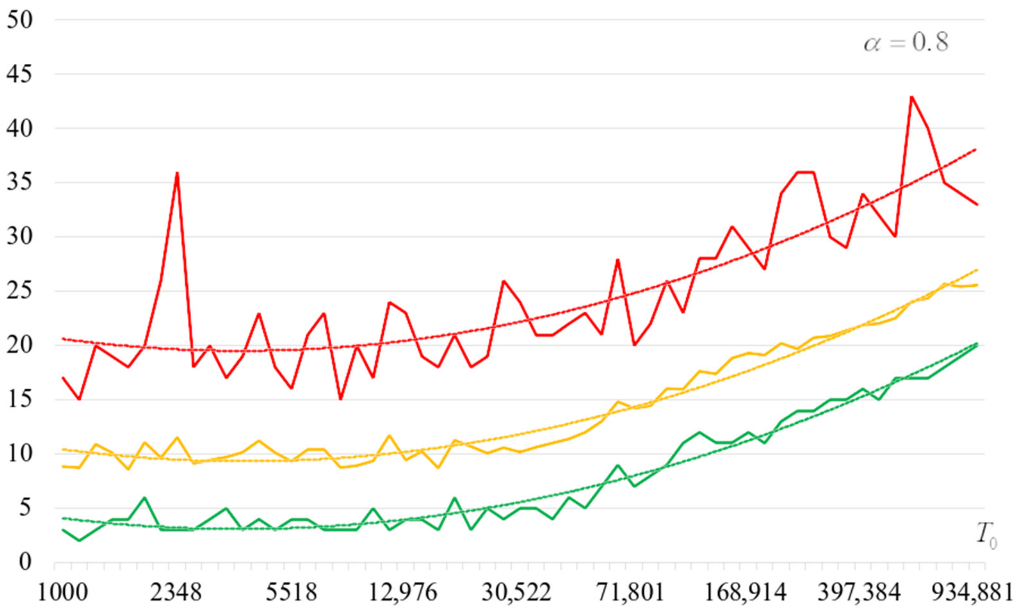

Figure 16. The dependencies of the number of iterations of the outer loop (until one of the criteria for stopping the algorithm is fulfilled) if the algorithm is successfully started, $\alpha = 0.8$.

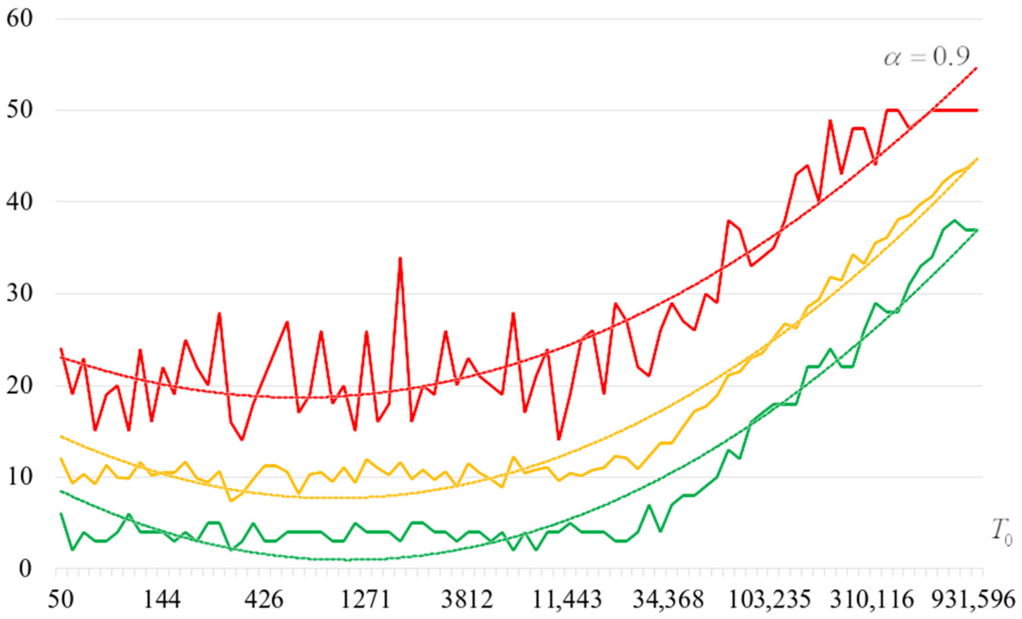

Figure 17. The dependencies of the number of iterations of the outer loop (until one of the criteria for stopping the algorithm is fulfilled) if the algorithm is successfully started, $\alpha = 0.9$.

Analysis of the dependencies shown in Figures 2–5 shows that when the initial temperature T_0 increases, the probability of forming the target S-box also increases. However, the average generation time does not decrease. This can be clearly seen in Figures 6–9. For each value of α, we have the "optimal" value of the initial temperature T_0, at which the generation time is minimized. This conclusion is also confirmed by Figures 10–17, where we see that all iteration number curves can be optimized by the value of the initial temperature T_0.

The final dependencies of the total number of iterations (external and internal cycle) are shown in Figures 18–21.

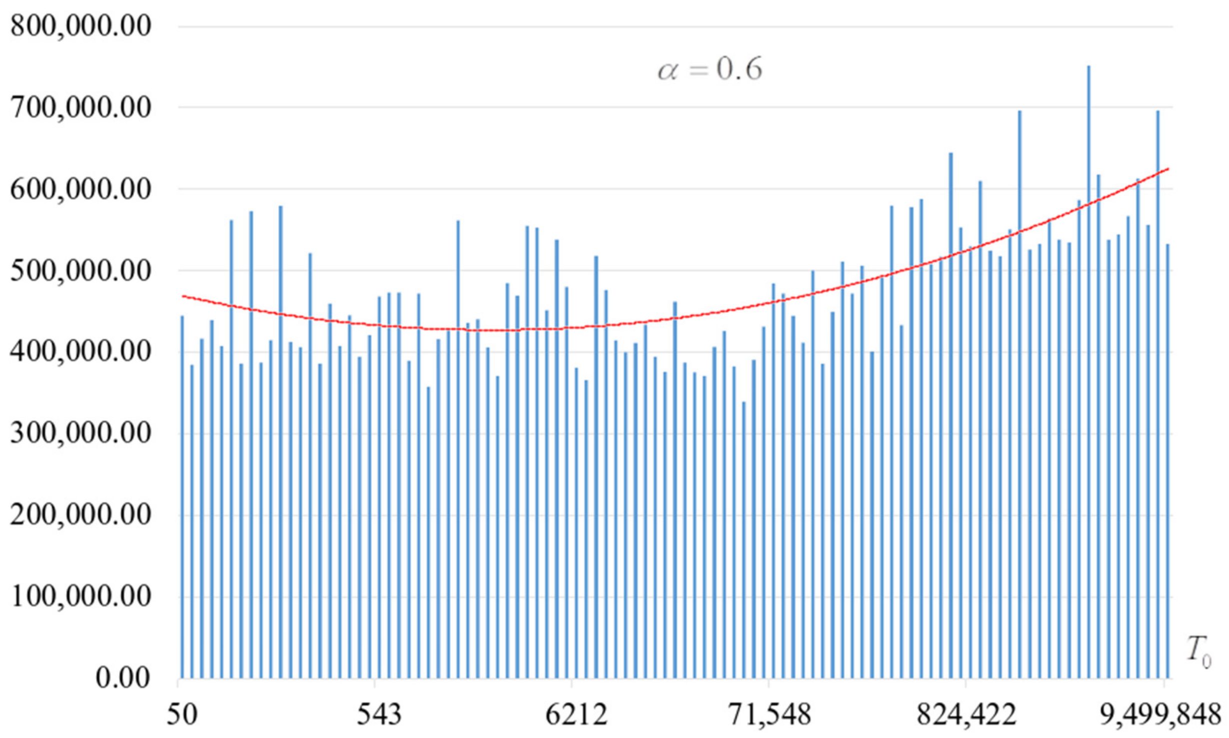

Figure 18. The summary number of the arithmetic mean number of iterations when searching for the target S-box, $\alpha = 0.6$.

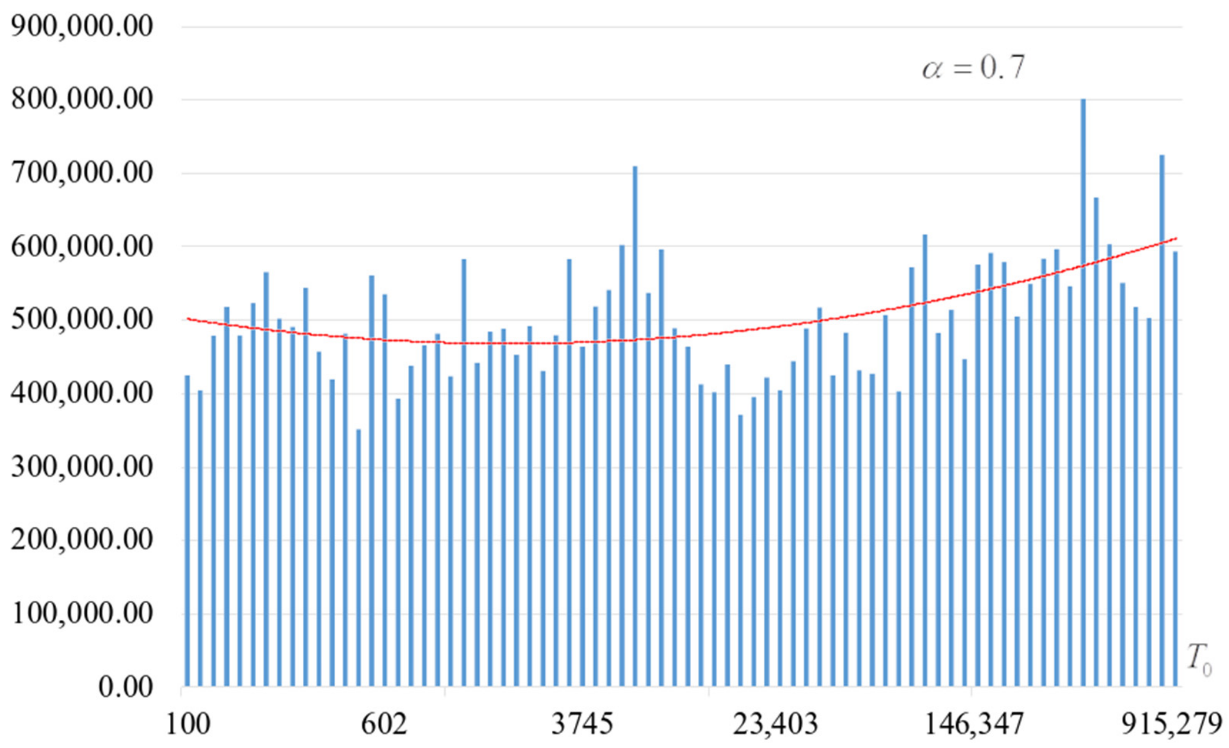

Figure 19. The summary number of the arithmetic mean number of iterations when searching for the target S-box, $\alpha = 0.7$.

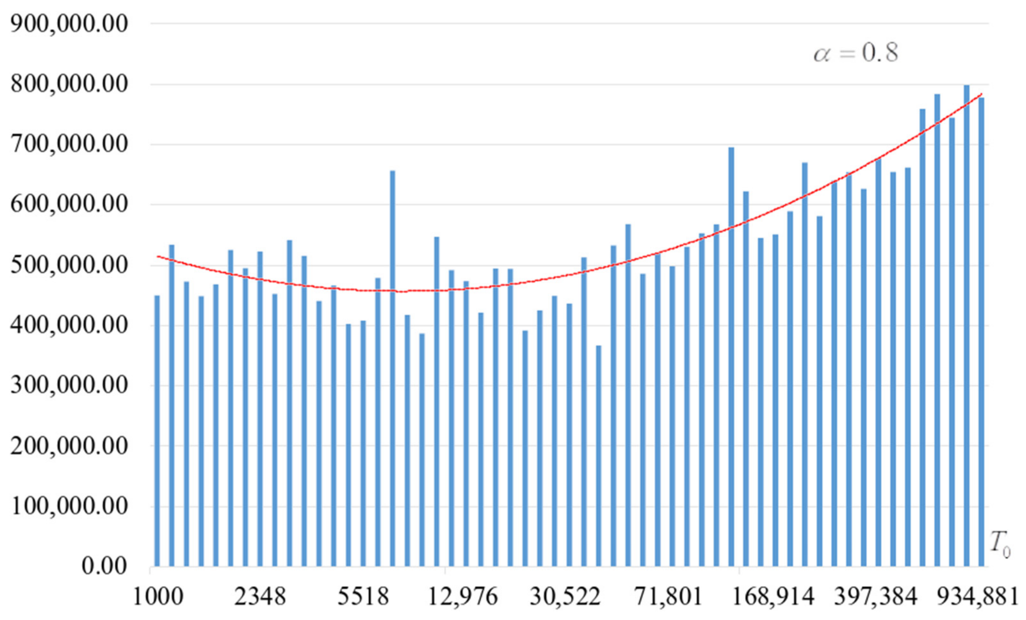

Figure 20. The summary number of the arithmetic mean number of iterations when searching for the target S-box, $\alpha = 0.8$.

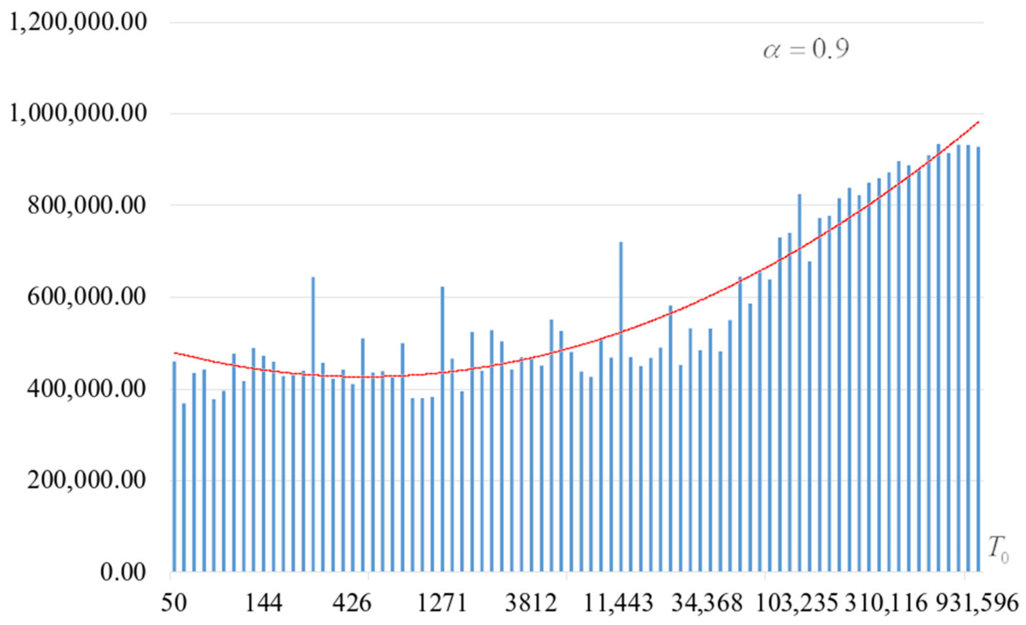

Figure 21. The summary number of the arithmetic mean number of iterations when searching for the target S-box, $\alpha = 0.9$.

The analysis of the dependencies shown in Figures 18–21 allows us to draw the following conclusions:

- As the value of T_0 grows, the total number of iterations is almost the same until a certain value (T_0 = 30,000…70,000, depending on α) and then begins to grow rapidly, which causes a significant increase in the time to execute each run of the algorithm.
- If the α is increased, the total number of search iterations decreases, i.e., substitution generation is performed faster.

6. Discussion

At small values of the initial temperature, the probability of making a worsening decision is very small, and therefore the simulated annealing algorithm behaves like a normal algorithm for finding a local minimum and, accordingly, has the same probability value of forming the target S-box and the average search time.

As the initial temperature increases, the probability of making worsening decisions increases, leading to an exit from the current state, which on the one hand may be an unnecessary local minimum and on the other hand may be one of the acceptable decisions that can lead to the formation of the target S-box. Analysis of the results indicates that the arithmetic mean value of the nonlinearity N_f is reached approximately at the outer loop iteration, which corresponds to the current temperature of the found minimum (T_0 =20,000 . . . 40,000).

A higher temperature leads to so-called *non-productive deteriorations*, i.e., deteriorations that lead to a permanent rollback of the found solution to the deteriorated state. Therefore, more iterations that are performed under nonproductive deteriorations can also be referred to *nonproductive iterations*, i.e., those that do not lead to improvement of the overall state of the system.

From the analysis of N_f results, it can be seen that in the right part, the arithmetic mean values of N_f reach the values corresponding to the left part after the first iteration of the outer loop only after approximately the number of iterations that lead the current temperature to the values of the found minimum (T_0 =20,000 . . . 40,000).

The search time for the target S-box also changes. Starting from small values of T_0 with a gradual increase, the search time decreases and at the end of the middle stage can be 1.5–2 times less than the value. Then, given the significant amount of non-productive degradation, the search time increases significantly. The higher the value of T_0, the greater the amount of nonproductive degradation, and the higher the value of α, the longer it lasts.

If the initial temperature is high or the rate of its decrease is low, a significant number of external cycles is needed to stabilize the system in some local minimum. If the maximum number of external cycles is insufficient, the algorithm may not find a local minimum, which leads to a significant decrease in the number of solutions found or to their complete absence.

The initial temperature at which the probability of finding the target S-box is maximal and no unproductive iterations are observed will be called the *optimal temperature* (labeled as T_0^{opt}). The found minimum of the average time of formation of the target S-box corresponds to the initial temperature interval T_0^{opt} =20,000 . . . 40,000. As the parameter α increases, the minimum shifts toward a smaller value of T_0.

To increase the accuracy of the values obtained, the number of runs for each temperature was increased to 1000. For an acceptable test time, the range of values of T_0 = 12,500 $-$ 37,500 was reduced, and only 11 values of T_0 were tested with three values of $\alpha = 0.85; 0.9; 0.95$ (testing was performed for 77 h). The results of the probability of formation of the target S-box and the average time of formation are shown in Figures 22 and 23.

According to the given data, with the chosen parameters (k_{out} = 50, k_{int} = 650, k_{froz} = 5, K_{THREAD} = 30), the best results are obtained at $\alpha = 0.95$ and T_0 = 20,000. The probability of finding the target S-box (from $N_f = 104$) is 56.4%, and the average search time is 14.2 s.

To compare the results with other known implementations of the SA algorithm, Table 2 gives estimates of the difficulty of finding the target S-box (with nonlinearity N_f). The "-" marks in Table 2 indicate cases with indeterminate indicators.

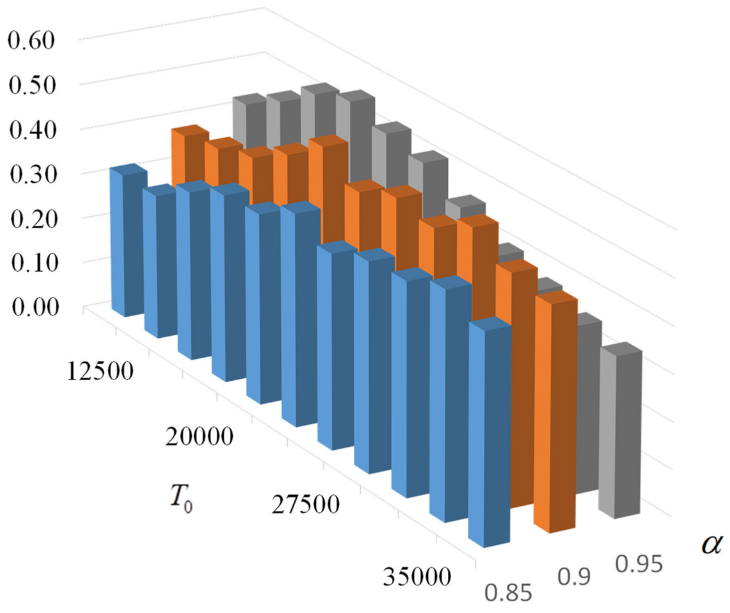

Figure 22. Probability of the target S-box generation when $\alpha = 0.85$; 0.9; 0.95 and $k_{int} = 650$ (each point is the average of 1000 tests).

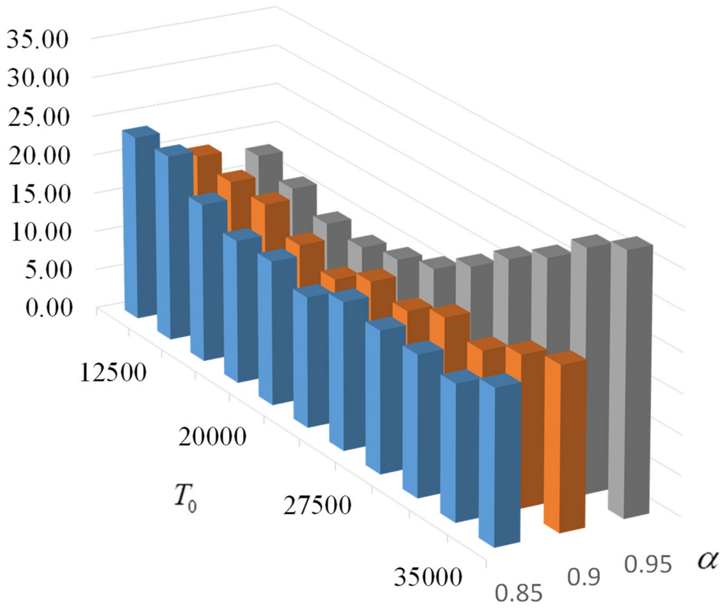

Figure 23. Average time (s) of the target S-box generation when $\alpha = 0.85$; 0.9; 0.95 and $k_{int} = 650$ (each point is the average of 1000 tests).

Table 2. Comparison of the results obtained on the generation of bijective 8-bit S-boxes (for different implementations of the SA algorithm).

	SA [11], SA [37]	SA [14]	SA [12]	Our Work
The highest value of N_f obtained in the found S-box	102	92	104	104
S-box generation probability	1/200 = 0.5%	-	-	56.4%
S-box generation (search) time	-	-	-	14.2 s
Generation complexity (number of search iterations)	-	-	3,000,000 30,000,000	450,000

7. Conclusions

We were able to significantly reduce the computational complexity of substitution generation using SA. In addition, we have also significantly increased the probability of generating the target S-boxes with a nonlinearity score of 104.

Based on the results of our studies, we conclude that the simulated annealing method does a good job of finding the target (i.e., with specified properties) S-box. If the algorithm parameters are well chosen, the probability of finding an S-box with nonlinearity $N_f = 104$ T will be almost unity.

However, a 100% probability of finding the target S-box is not the optimal path in terms of time spent searching. Introducing additional constraints reduces the time spent on each attempt but also reduces the probability of finding the target S-box in each attempt. Therefore, the search results using the simulated annealing method are very sensitive to all input search parameters, and their optimization is a very time-consuming process. The influence of input parameters of simulated annealing method on the search result of target S-box was investigated. Based on the results of the study, the comparative characteristics of the search time and the internal states of the algorithm are presented, and optimization by the search time minimization criterion was carried out. With the chosen algorithm parameters (k_{out} = 50, k_{int} = 650, k_{froz} = 5, K_{THREAD} = 30), the best results were obtained with $\alpha = 0.95$ and T_0 = 20,000. In this case, the probability of finding the target S-box (from $N_f = 104$) is 56.4%, and the average search time is 14.2 s. The algorithm requires about 450,000 search iterations on average. As the number of internal iterations increases, the probability of detecting the target S-box increases to 97%. This is the best known result of applying the SA algorithm to generate bijective 8-bit S-boxes.

Author Contributions: Conceptualization and methodology, A.K.; formal analysis, investigation, L.W.; N.P.; software, L.H.; software and validation, S.K.; writing—review and editing, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

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.

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. Kuznetsov, A.A.; Potii, O.V.; Poluyanenko, N.A.; Gorbenko, Y.I.; Kryvinska, N. *Stream Ciphers in Modern Real-Time IT Systems: Analysis, Design and Comparative Studies*; Studies in Systems, Decision and Control; Springer International Publishing: New York, NY, USA, 2022; ISBN 978-3-030-79769-0.
3. Kuznetsov, O.; Potii, O.; Perepelitsyn, A.; Ivanenko, D.; Poluyanenko, N. Lightweight Stream Ciphers for Green IT Engineering. In *Green IT Engineering: Social, Business and Industrial Applications*; Kharchenko, V., Kondratenko, Y., Kacprzyk, J., Eds.; Studies in Systems, Decision and Control; Springer International Publishing: Cham, Switzerland, 2019; Volume 171, pp. 113–137, ISBN 978-3-030-00252-7.
4. Shannon, C.E. Communication Theory of Secrecy Systems. *Bell Syst. Tech. J.* **1949**, *28*, 656–715. [CrossRef]
5. Rubinstein-Salzedo, S. *Cryptography*; Springer Undergraduate Mathematics Series; Springer International Publishing: Cham, Switzerland, 2018; ISBN 978-3-319-94817-1.
6. Courtois, N.T.; Pieprzyk, J. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In *Advances in Cryptology—ASIACRYPT 2002*; Zheng, Y., Ed.; Springer: Berlin/Heidelberg, Germany, 2002; pp. 267–287.
7. Courtois, N.T.; Bard, G.V. Algebraic Cryptanalysis of the Data Encryption Standard. In *Cryptography and Coding*; Galbraith, S.D., Ed.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 152–169.
8. 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.
9. Bard, G.V. *Algebraic Cryptanalysis*; Springer US: Boston, MA, USA, 2009; ISBN 978-0-387-88756-2.

10. Nover, H. *Algebraic Cryptanalysis of Aes: An Overview*; University of Wisconsin: Madison, WI, USA, 2005.
11. 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.
12. 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.
13. 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]
14. 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]
15. Delahaye, D.; Chaimatanan, S.; Mongeau, M. *Simulated Annealing: From Basics to Applications*; Springer: Berlin/Heidelberg, Germany, 2019; Volume 272, p. 1, ISBN 978-3-319-91086-4.
16. Eremia, M.; Liu, C.-C.; Edris, A.-A. Heuristic Optimization Techniques. In *Advanced Solutions in Power Systems: HVDC, FACTS, and Artificial Intelligence*; IEEE: Piscataway, NJ, USA, 2016; pp. 931–984, ISBN 978-1-119-17533-9.
17. Beth, T.; Ding, C. On Almost Perfect Nonlinear Permutations. In *Advances in Cryptology—EUROCRYPT '93*; Helleseth, T., Ed.; Springer: Berlin/Heidelberg, Germany, 1994; pp. 65–76.
18. Nyberg, K. Differentially Uniform Mappings for Cryptography. In *Advances in Cryptology—EUROCRYPT '93*; Helleseth, T., Ed.; Springer: Berlin/Heidelberg, Germany, 1994; pp. 55–64.
19. Clark, A.J. Optimisation Heuristics for Cryptology. Ph.D. Thesis, Queensland University of Technology, Brisbane, Australia, 1998.
20. Kose, U.; Kose, U.; Guraksin, G.E.; Deperlioglu, O. (Eds.) *Nature-Inspired Intelligent Techniques for Solving Biomedical Engineering Problems*; Advances in Bioinformatics and Biomedical Engineering; IGI Global: Hershey, PA, USA, 2018; ISBN 978-1-5225-4769-3.
21. Information Resources Management Association. *Nature-Inspired Computing: Concepts, Methodologies, Tools, and Applications*; IGI Global: Hershey, PA, USA, 2017; ISBN 978-1-5225-0788-8.
22. Korte, B.; Vygen, J. *Combinatorial Optimization: Theory and Algorithms*, 6th ed.; Springer: Berlin/Heidelberg, Germany, 2018.
23. Oliva, D.; Houssein, E.H.; Hinojosa, S. (Eds.) *Metaheuristics in Machine Learning: Theory and Applications*; Studies in Computational Intelligence; Springer International Publishing: Cham, Switzerland, 2021; Volume 967, ISBN 978-3-030-70541-1.
24. Zamani, H.; Nadimi-Shahraki, M.H.; Gandomi, A.H. Starling Murmuration Optimizer: A Novel Bio-Inspired Algorithm for Global and Engineering Optimization. *Comput. Methods Appl. Mech. Eng.* **2022**, *392*, 114616. [CrossRef]
25. Li, Y.; Carabelli, S.; Fadda, E.; Manerba, D.; Tadei, R.; Terzo, O. Machine Learning and Optimization for Production Rescheduling in Industry 4.0. *Int. J. Adv. Manuf. Technol.* **2020**, *110*, 2445–2463. [CrossRef]
26. Zamani, H.; Nadimi-Shahraki, M.H.; Gandomi, A.H. QANA: Quantum-Based Avian Navigation Optimizer Algorithm. *Eng. Appl. Artif. Intell.* **2021**, *104*, 104314. [CrossRef]
27. Yu, V.F.; Susanto, H.; Jodiawan, P.; Ho, T.-W.; Lin, S.-W.; Huang, Y.-T. A Simulated Annealing Algorithm for the Vehicle Routing Problem With Parcel Lockers. *IEEE Access* **2022**, *10*, 20764–20782. [CrossRef]
28. Tang, O. Simulated Annealing in Lot Sizing Problems. *Int. J. Prod. Econ.* **2004**, *88*, 173–181. [CrossRef]
29. Tesar, P. A New Method for Generating High Non-Linearity S-Boxes. *Radioengineering* **2010**, *19*, 23–26.
30. Picek, S.; Cupic, M.; Rotim, L. A New Cost Function for Evolution of S-Boxes. *Evol. Comput. Winter* **2016**, *24*, 695–718. [CrossRef] [PubMed]
31. 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]
32. Freyre Echevarría, A.; Martínez Díaz, I. A New Cost Function to Improve Nonlinearity of Bijective S-Boxes. 2020; *preprint*.
33. Kuznetsov, A.; Poluyanenko, N.; Kandii, S.; Zaichenko, Y.; Prokopovich-Tkachenko, D.; Katkova, T. WHS Cost Function 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. 434–438.
34. Chen, G. A Novel Heuristic Method for Obtaining S-Boxes. *Chaos Solitons Fractals* **2008**, *36*, 1028–1036. [CrossRef]
35. 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.
36. McLaughlin, J. Applications of Search Techniques to Cryptanalysis and the Construction of Cipher Components. Ph.D. Thesis, University of York, Heslington, UK, 2012.
37. Ivanov, G.; Nikolov, N.; Nikova, S. Cryptographically Strong S-Boxes Generated by Modified Immune Algorithm. In *Cryptography and Information Security in the Balkans*; Pasalic, E., Knudsen, L.R., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 31–42.