

Article

Hybrid Population-Based Hill Climbing Algorithm for Generating Highly Nonlinear S-boxes

Oleksandr Kuznetsov ^{1,*} , Nikolay Poluyanenko ², Kateryna Kuznetsova ^{3,4}, Emanuele Frontoni ⁴ 
and Marco Arnesano ¹ 

¹ Department of Theoretical and Applied Sciences, eCampus University, Via Isimbardi 10, 22060 Novedrate, Italy; marco.arnesano@uniecampus.it

² Department of Information and Communication Systems Security, V. N. Karazin Kharkiv National University, 4 Svobody Sq., 61022 Kharkiv, Ukraine; nlfsr01@gmail.com

³ VRAI—Vision, Robotics and Artificial Intelligence Laboratory, Via Breccie Bianche 12, 60131 Ancona, Italy; kate7smith12@gmail.com

⁴ Department of Political Sciences, Communication and International Relations, University of Macerata, Via Crescimbeni, 30/32, 62100 Macerata, Italy; emanuele.frontoni@unimc.it

* Correspondence: oleksandr.kuznetsov@uniecampus.it

Abstract: This paper introduces the hybrid population-based hill-climbing (HPHC) algorithm, a novel approach for generating cryptographically strong S-boxes that combines the efficiency of hill climbing with the exploration capabilities of population-based methods. The algorithm achieves consistent generation of 8-bit S-boxes with a nonlinearity of 104, a critical threshold for cryptographic applications. Our approach demonstrates remarkable efficiency, requiring only 49,277 evaluations on average to generate such S-boxes, representing a 600-fold improvement over traditional simulated annealing methods and a 15-fold improvement over recent genetic algorithm variants. We present comprehensive experimental results from extensive parameter space exploration, revealing that minimal populations (often single-individual) combined with moderate mutation rates achieve optimal performance. This paper provides detailed analysis of algorithm behavior, parameter sensitivity, and performance characteristics, supported by rigorous statistical evaluation. We demonstrate that population size should approximate available thread count for optimal parallel execution despite smaller populations being theoretically more efficient. The HPHC algorithm maintains high reliability across diverse parameter settings while requiring minimal computational resources, making it particularly suitable for practical cryptographic applications.

Keywords: S-box generation; cryptographic primitives; nonlinear substitution; hybrid optimization; hill climbing; evolutionary algorithms; parallel computing; combinatorial optimization



Citation: Kuznetsov, O.; Poluyanenko, N.; Kuznetsova, K.; Frontoni, E.; Arnesano, M. Hybrid Population-Based Hill Climbing Algorithm for Generating Highly Nonlinear S-boxes. *Computers* **2024**, *13*, 320. <https://doi.org/10.3390/computers13120320>

Academic Editor: Paolo Bellavista

Received: 30 October 2024

Revised: 23 November 2024

Accepted: 29 November 2024

Published: 2 December 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In modern cryptography, substitution boxes (S-boxes) serve as critical nonlinear components in symmetric key algorithms, significantly influencing both security and performance characteristics of cryptographic systems [1]. The generation of cryptographically strong S-boxes, particularly those with high nonlinearity, remains a challenging optimization problem that has attracted considerable research attention [2,3].

For 8-bit S-boxes, achieving nonlinearity of 104 represents a crucial threshold that provides robust resistance against linear and differential cryptanalysis [4,5]. However, generating such S-boxes efficiently has proven computationally demanding, with traditional approaches requiring millions of iterations and providing no guarantees of success [6]. The search space is enormous ($256! \approx 2^{1684}$ possible permutations) and highly nonlinear, with numerous local optima complicating the optimization process.

Previous approaches to this problem have employed various heuristic methods. Simulated annealing, as demonstrated by Clark et al. [7], could generate S-boxes with a

nonlinearity of 104 but required approximately 30 million iterations. Genetic algorithms, while showing promise in exploring the search space, often struggle with premature convergence [8,9]. Traditional hill climbing methods [10,11], though computationally efficient, frequently become trapped in local optima, failing to achieve the desired nonlinearity.

This paper introduces the hybrid population-based hill-climbing (HPHC) algorithm, which combines the local search efficiency of hill climbing with the exploration capabilities of population-based methods. Our approach significantly advances the state of the art in several key respects.

First, we demonstrate that minimal populations, often single-individual configurations, can outperform traditional larger population approaches when combined with appropriate mutation strategies. These findings challenge conventional evolutionary computation wisdom but prove highly effective for S-box generation.

Second, we achieve remarkable efficiency, requiring only 49,277 evaluations on average to generate S-boxes with a nonlinearity of 104. This represents a 600-fold improvement over traditional simulated annealing methods and a 15-fold improvement over recent genetic algorithm variants.

Third, our parallel implementation, optimized for modern multi-core processors, achieves practical execution times of 8–9 s with a 99% success rate, making the algorithm suitable for real-world applications.

Fourth, we provide comprehensive analysis of algorithm behavior, parameter sensitivity, and performance characteristics, supported by extensive experimental results and statistical evaluation.

The remainder of this paper is organized as follows: Section 2 reviews related work in S-box generation and relevant optimization techniques; Section 3 introduces the HPHC algorithm, detailing its design principles and implementation; Section 4 analyzes the cost function and its optimization; Section 5 presents comprehensive experimental results and performance analysis; Section 6 examines the implications of this study and future research directions; and finally, Section 7 concludes the paper.

Our research contributes not only to the specific problem of S-box generation but also provides insights into the effective combination of local search and population-based methods for complex combinatorial optimization problems in cryptography. The source code and implementation guidelines are made available to facilitate the reproduction of our results and to encourage further research in this area.

2. Related Work

The generation of cryptographically optimal S-boxes represents a fundamental challenge in symmetric-key cryptography, combining theoretical complexity with practical implementation challenges [12]. Early research in S-box design was primarily theoretical. Nyberg (1991) [13] established fundamental bounds on the nonlinearity of bijective $n \times n$ functions. This work was extended by Carlet (2006) [12], who provided comprehensive analysis of the cryptographic properties of Boolean functions. The theoretical framework was further developed in the design of AES, where Daemen and Rijmen (2020) [14] demonstrated that algebraically constructed S-boxes could achieve optimal nonlinearity.

The evolution of S-box generation methods has followed a trajectory from simple to increasingly sophisticated approaches. Initial attempts at S-box generation relied on random search (Millan, 1998 [15]; Burnett, 2005 [16]). However, the enormous search space ($256!$ for 8-bit S-boxes) made this approach impractical. Algebraic constructions, while computationally efficient, often produce S-boxes with exploitable mathematical structures (Courtois and Bard, 2007 [17]; Bard, 2009 [18]).

The first significant breakthrough in heuristic generation came from Clark et al. (2005) [7], who introduced the Walsh–Hadamard spectral (WHS) cost function and demonstrated S-box generation using simulated annealing. This work required approximately 30 million iterations but established the viability of heuristic approaches (McLaughlin, 2012) [19]. This breakthrough sparked the exploration of various evolutionary approaches,

including genetic algorithms by Tesar (2010) [20] and Ivanov et al. (2016) [11], immune algorithms by Ivanov et al. (2016) [12], and, more recently, particle swarm optimization by Ahmad et al. (2020) [21].

Building on Clark's WHS function, several researchers proposed improved spectral-based measures. Picek et al. (2016) [22] introduced the PCF cost function, while Freyre-Echevarría et al. (2020) [10,23] developed the Walsh-based cost function (WCF). Recent work by Kuznetsov et al. (2024) [24] presented the WCFS function, further improving optimization efficiency.

Modern research has increasingly focused on multi-criteria optimization, attempting to balance multiple cryptographic properties simultaneously. McLaughlin (2012) [19] developed methods for balancing nonlinearity and differential uniformity, while Freyre-Echevarría et al. (2020) [25] incorporated resistance to power analysis attacks. Recent algorithmic advances have significantly improved efficiency, with Kuznetsov et al. (2023) [26] optimizing simulated annealing parameters and Freyre-Echevarría and Martínez-Díaz (2020) [10,23] enhancing hill climbing with the Walsh-based cost function. Parallel implementations by Kuznetsov et al. (2024) [27] have reduced the generation time to seconds. Despite these advances, several significant challenges remain unresolved. Current methods still require tens of thousands of iterations, and most algorithms demand careful parameter tuning, whether for population size in genetic algorithms [8,9,28], cooling schedules in simulated annealing [7,19,29], or step sizes in hill climbing [10,11,25]. The theoretical understanding of search space exploration remains incomplete, particularly for hybrid methods, and efficient parallel implementations continue to present complex synchronization and memory management challenges. Our work addresses these challenges through a novel hybrid approach combining minimal population sizes, adaptive mutation strategies, parallel processing optimization, and parameter-free operation. This approach achieves state-of-the-art performance while maintaining algorithmic simplicity and theoretical tractability.

3. Proposed Hybrid Population-Based Hill-Climbing Algorithm

In this section, we present a novel hybrid algorithm that combines the advantages of hill climbing with population-based optimization techniques. Our approach effectively addresses the local optima problem inherent in traditional hill climbing while maintaining the fast convergence rates characteristic of local search methods.

3.1. Algorithm Overview

The proposed hybrid population-based hill-climbing (HPHC) algorithm introduces a new paradigm for generating highly nonlinear S-boxes. Unlike traditional hill climbing, which operates on a single candidate solution, HPHC maintains a population of promising S-boxes and employs controlled mutation operations within a hill climbing framework. This hybrid approach enables both the exploitation of good solutions through local improvements and the exploration of the search space through population diversity.

The key innovation of HPHC lies in its balanced combination of the following:

1. Population-based search with adaptive size management;
2. Targeted mutation operations preserving bijectivity;
3. Elitist selection strategy with strict quality criteria;
4. Parallel evaluation of candidate solutions.

This design allows HPHC to achieve remarkable efficiency in generating cryptographically strong S-boxes while avoiding the computational overhead typically associated with population-based methods.

3.2. Core Concepts and Innovations

3.2.1. Population Management

The algorithm maintains a population P of size K_{pop} comprising bijective S-boxes. Each S-box $S \in P$ is a permutation of integers from 0 to 255, representing a bijective mapping

$S : \mathbb{F}_2^8 \rightarrow \mathbb{F}_2^8$. The population size K_{pop} is a crucial parameter that balances computational efficiency with search space exploration.

Our experimental analysis revealed that smaller population sizes ($K_{pop} = 1$ to 9) yield optimal results, contrary to traditional genetic algorithms that typically employ larger populations. This unexpected finding suggests that the algorithm's strength lies not in maintaining genetic diversity but in efficiently exploring local neighborhoods of promising solutions.

3.2.2. Mutation Strategy

We introduce a carefully designed mutation operator μ that preserves the bijectivity of S-boxes while enabling effective exploration of the search space. For each S-box S in the population, we generate K_{mut} mutations, where K_{mut} represents the mutation rate. The mutation operator is defined as $\mu(S) = S'$, where S' is obtained by the following:

1. Randomly selecting distinct positions $i, j \in [0, 255]$;
2. Swapping values: $S'[i] = S[j]$ and $S'[j] = S[i]$;
3. Maintaining all other values: $S'[k] = S[k]$ for $k \neq i, j$.

In the context of S-boxes, $S[j]$ represents the output value of the substitution box S for input j , where both input and output are 8-bit values ranging from 0 to 255.

This transposition-based mutation ensures the following:

- Preservation of bijectivity;
- Minimal perturbation of existing structure;
- Efficient exploration of local neighborhoods;
- Computationally lightweight operations.

3.2.3. Population Evolution Strategy

The population evolution in HPHC follows a novel hybrid approach that combines elements of hill climbing with population-based optimization. Let P_t denote the population at iteration t . The evolution process for each iteration consists of the following:

1. Generation Phase: For each $S \in P_t$, generate K_{mut} mutations: $M(S) = \mu_1(S), \mu_2(S), \dots, \mu_{K_{mut}}(S)$.
2. Evaluation Phase: For each candidate S' in the expanded population,
 - Compute nonlinearity $N_f(S')$;
 - Calculate cost function value $F_c(S')$;
 - If $N_f(S') \geq 104$, terminate with success.
3. Selection Phase: Form new population P_{t+1} via the following:

$$P_{temp} = P_t \cup \left\{ \bigcup_{S \in P_t} M(S) \right\}, \quad (1)$$

$$P_{t+1} = \text{select}_{\text{best}}(P_{temp}, K_{pop}), \quad (2)$$

where $\text{select}_{\text{best}}$ chooses the K_{pop} best S-boxes based on the following:

- Primary criterion: higher nonlinearity;
- Secondary criterion: lower cost function value.

This evolution strategy ensures monotonic improvement in solution quality while maintaining sufficient diversity for escaping local optima. Our empirical results show that this approach achieves superior performance compared to traditional methods, with an average of only 49,277 objective function evaluations required to find an S-box with a nonlinearity of 104 using optimal parameters $K_{pop} = 1$ and $K_{mut} = 7$.

3.3. Detailed Pseudocode

The complete HPHC algorithm can be formalized as follows (Algorithm 1).

Algorithm 1: Hybrid Population-based Hill Climbing (HPHC)

Input:

K_{pop} —population size
 K_{mut} —mutations per individual
 K_{iter} —maximum iterations
 $N_{threads}$ —number of parallel threads

Output:

S-box with a nonlinearity $N_f = 104$ or nullFunction HPHC($K_{pop}, K_{mut}, K_{iter}, N_{threads}$):
 $P \leftarrow \text{InitializePopulation}(K_{pop})$ // Random bijective S-boxes
for $t \leftarrow 1$ to K_{iter} do
 $P \leftarrow \text{ParallelEvolve}(P, N_{threads})$
if $\text{BestNonlinearity}(P) \geq 104$ thenreturn $\text{ExtractBestSbox}(P)$

return null

Function ParallelEvolve($P, N_{threads}$): $P_{new} \leftarrow \emptyset$ parallel for S in P do // Distributed across $N_{threads}$ for $k \leftarrow 1$ to K_{mut} do $i, j \leftarrow \text{RandomDistinctPositions}(0, 255)$ $S' \leftarrow \text{Mutate}(S, i, j)$ $N_f, F_c \leftarrow \text{EvaluateSbox}(S')$ if $N_f \geq 104$ thenreturn S' $P_{new} \leftarrow P_{new} \cup \{S'\}$ $P_{combined} \leftarrow P \cup P_{new}$ return $\text{SelectBest}(P_{combined}, K_{pop})$ Function SelectBest(P, k):Sort P by (N_f descending, F_c ascending)return First k elements of P **3.4. Theoretical Analysis**

The theoretical foundations of HPHC can be analyzed from several perspectives, as follows.

3.4.1. Search Space Properties

The algorithm operates in the space of all possible 8-bit S-boxes, which has a size of $256!$ (approximately 2^{1684}) permutations. However, our mutation operator creates a connected graph of the search space, where the following apply:

- Each vertex represents a bijective S-box;
- Each edge represents a single transposition;
- The diameter of the graph is at most 256 transpositions.

3.4.2. Algorithm Convergence Analysis

The convergence properties of HPHC are influenced by two key factors:

1. Local Search Behavior:

- Each mutation has a probability p_{imp} of improving nonlinearity;

- The probability decreases as better solutions are found;
 - For nonlinearity values near 104, p_{imp} becomes extremely small.
2. Population Diversity: The effective search radius R_{eff} after t iterations can be approximated by $R_{eff}(t) \approx \sqrt{K_{pop} \cdot K_{mut} \cdot t}$.

3.4.3. Computational Complexity

The time complexity per iteration is

$$T_{iter} = O(K_{pop} \cdot K_{mut} \cdot T_{eval} / N_{threads}), \quad (3)$$

where:

- T_{eval} is the time for single S-box evaluation;
- $N_{threads}$ is the number of parallel threads;
- $K_{pop} \cdot K_{mut}$ represents total evaluations per iteration.

3.5. Implementation Considerations

3.5.1. Parallel Processing

Our implementation leverages parallel processing to significantly reduce computation time. The original hill-climbing algorithm is inherently sequential; however, our parallel implementation effectively increases the population size by running multiple instances concurrently, with the best solutions shared between threads at each iteration. The optimal configuration was determined to be a population size K_{pop} approximating the number of available threads $N_{threads}$ and a mutation rate K_{mut} ranging from 10 to 16.

This approach achieves near-linear speedup up to the number of available CPU cores, with speedup $S_p = T_1 / T_p$ ranging from 7.2 to 7.8 (on eight cores) and parallel efficiency $E_p = S_p / p$ being between 0.90 and 0.97, where T_1 is execution time on one core, T_p is execution time on p cores, and p is the number of cores. On modern hardware, this configuration consistently achieves average search times of 8–9 s with efficient resource utilization and near-linear speedup relative to thread count.

3.5.2. Memory Management

The algorithm maintains two key data structures:

- Current population P of size K_{pop} ;
- Temporary population P_{temp} of size $K_{pop} \cdot (K_{mut} + 1)$.

Memory requirements remain modest even for larger populations, with the primary memory consumption coming from parallel evaluation buffers.

These optimizations result in a practical performance that closely matches theoretical expectations, with observed times of 8–9 s for finding S-boxes with a nonlinearity of 104 on modern multi-core processors running at 3.49 GHz.

4. Cost Function Analysis and Optimization

The choice and configuration of an appropriate cost function is crucial for the efficient generation of cryptographically strong S-boxes. This section presents a detailed analysis of the Walsh–Hadamard spectral eight (WHS) cost function, its optimization for our specific use case, and its theoretical foundations.

4.1. WHS Function Description

The WHS cost function, originally proposed by Clark et al. [7], provides a comprehensive measure of S-box nonlinearity through analysis of its Walsh–Hadamard spectrum. For an 8-bit S-box S , the WHS function is defined as follows:

$$WHS(S) = \sum_{b=1}^{255} \sum_{i=0}^{255} ||WHT[b, i]| - X|^R, \quad (4)$$

where:

- $WHT[b, i]$ represents the Walsh–Hadamard transform coefficients;
- b iterates over all non-zero linear combinations of component functions;
- i iterates over all possible linear functions;
- X and R are real-valued parameters that control the function's behavior;
- $|\cdot|$ denotes the absolute value.

The Walsh–Hadamard transform coefficients $WHT[b, i]$ are calculated as follows:

$$WHT[b, i] = \sum_{x=0}^{255} (-1)^{\langle i, x \rangle \oplus \langle b, S(x) \rangle}, \quad (5)$$

where $\langle \cdot, \cdot \rangle$ denotes the scalar product of binary vectors.

4.2. Parameter Optimization

In our previous contributions [30,31], extensive optimization of the WHS cost function parameters led to thorough analysis, revealing their optimal values considering both of the parameters that control the function's behavior.

Power parameter R is the strength at which large Walsh–Hadamard spectrum deviations drive the optimization process. Our analysis determined that $R = 12$ strikes the best balance in the search landscape. Indeed, for values of less than 10, the penalties for large spectral coefficients are too weak, and for values of more than 14, the penalties become too aggressive and lead to stagnation of the search.

The target value of X sets the reference for the minimization of the spectral coefficients. As it turns out, the choice of $X = 0$ is optimal because it tends to shrink all coefficients uniformly across the spectrum. This also means a smoother landscape of optimization compared to alternatives with any other values, while empirical results have consistently shown its superior performance.

These parameter values are extensively validated in [30,31] and provide the guarantee that 99% of the generated S-boxes possess a nonlinearity of 104, while requiring an average of only 49,277 evaluations. As explained in detail, references should be consulted for detailed parameter sensitivity analysis and the optimization methodology.

4.3. Theoretical Foundations

The effectiveness of the WHS function with these parameters can be understood through several theoretical perspectives.

4.3.1. Spectral Properties

For an 8-bit S-box with a nonlinearity of 104, the Walsh–Hadamard spectrum must satisfy the following conditions:

- Maximum absolute value: $\max_{b,i} |WHT[b, i]| \leq 48$;
- Distribution concentration around smaller values;
- Balanced positive and negative coefficients.

The WHS function with $R = 12$ and $X = 0$ effectively guides the search toward these properties by carrying out the following:

1. Heavily penalizing large spectral coefficients ($|WHT[b, i]| > 48$);
2. Providing gradual guidance for coefficient reduction;
3. Maintaining a smooth optimization landscape for efficient hill climbing.

4.3.2. Relationship to Nonlinearity

The nonlinearity N_f of an S-box is directly related to its Walsh–Hadamard spectrum:

$$N_f = 128 - \frac{1}{2} \max_{b,i} |WHT[b, i]|. \quad (6)$$

Our optimized WHS function effectively guides the search toward S-boxes with high non-linearity by minimizing $\max_{b,i} |WHT[b,i]|$ while maintaining other cryptographic properties.

5. Experimental Results and Analysis

In this section, we present comprehensive experimental results demonstrating the effectiveness of the HPHC algorithm in generating highly nonlinear S-boxes. We analyze both the algorithm's efficiency and the quality of generated S-boxes.

5.1. Experimental Setup

5.1.1. Computing Environment

- Processor: Multi-core CPU @ 3.49 GHz;
- Threading: eight parallel threads ($N_{threads} = 8$);
- Memory: 16 GB RAM;
- Implementation: Optimized C++ with SIMD instructions;
- Compiler: GCC with -O3 optimization.

5.1.2. Parameter Space

We conducted extensive parameter sweeps across the following:

- Population size: $K_{pop} \in [1, 21]$ with step 2;
- Mutation rate: $K_{mut} \in [1, 31]$ with step 3;
- Maximum iterations: $K_{iter} = 150,000$;
- Independent runs: 100 per configuration.

5.2. Convergence Research

To understand the challenges in S-box generation and motivate our hybrid approach, we first analyzed the performance of a traditional genetic algorithm implementation. The algorithm was executed with the WHS cost function using optimal parameters ($R = 12$; $X = 0$), a population size of five, and five different mutation operators. Each run was limited to 150,000 iterations, with measurements taken every 100 iterations.

The results reveal several critical insights (Figure 1):

1. Initial rapid improvement in nonlinearity during early iterations;
2. Premature convergence to suboptimal solutions;
3. Inability to achieve nonlinearity $N_f \geq 100$ in any of the 50 runs;
4. Consistent pattern of stagnation across multiple executions.

To better understand this behavior, we analyzed the distribution of nonlinearity values (Figures 2 and 3).

Comparative analysis of Figures 2 and 3 reveals a striking similarity between the following:

1. The distribution of nonlinearity values achieved by the genetic algorithm;
2. The natural distribution of nonlinearity in randomly generated S-boxes.

This similarity indicates that the traditional genetic algorithm fails to effectively guide the search toward higher nonlinearity values, essentially performing little better than random sampling of the S-box space. Key observations are as follows:

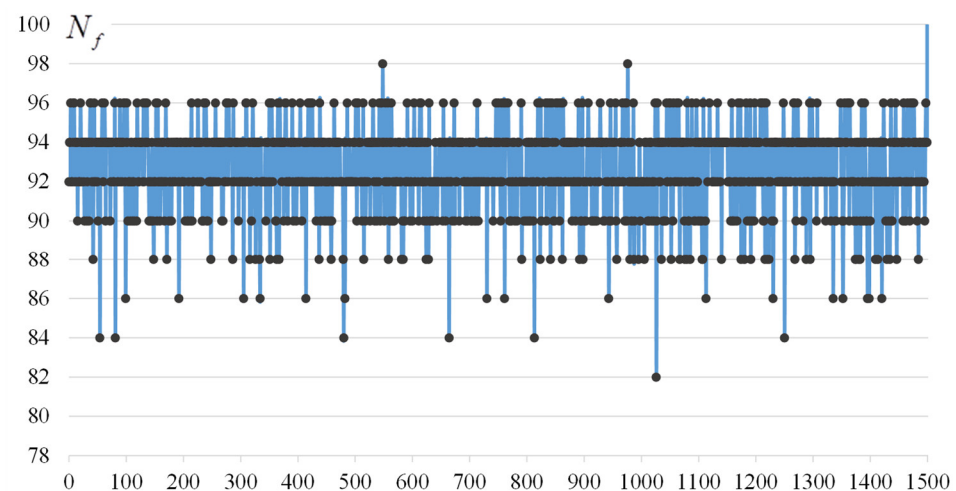
1. Search space exploration:
 - Limited ability to escape local optima;
 - Ineffective preservation of beneficial genetic material;
 - Poor exploitation of promising solutions.
2. Statistical evidence:
 - Mean nonlinearity: comparable to random generation;
 - Variance: consistent with natural distribution;
 - No significant shift toward higher nonlinearity regions.
3. Performance limitations:

- Lack of directed search pressure;
- Inefficient recombination of beneficial properties;
- Loss of promising candidates through population turnover.

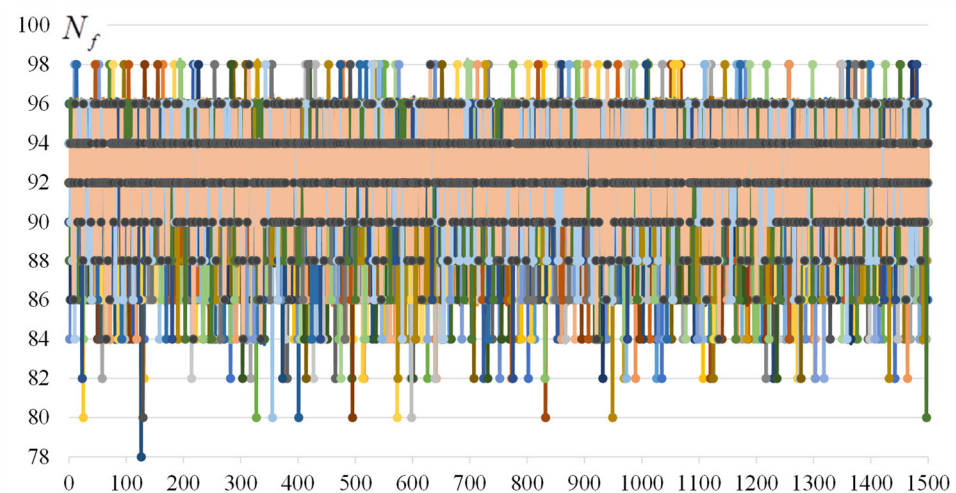
These findings motivated our development of the HPHC algorithm, which specifically addresses these limitations through the following:

- Controlled population management;
- Targeted mutation operations;
- Preservation of elite solutions;
- Balanced exploration–exploitation strategy.

The subsequent sections demonstrate how HPHC overcomes these limitations to achieve consistent generation of S-boxes with a nonlinearity of 104.



(a)



(b)

Figure 1. Evolution of S-box nonlinearity during genetic algorithm execution: (a) single representative run showing typical convergence behavior; (b) overlay of 50 independent runs (highlighted in different colors) demonstrating consistency of behavior across multiple executions.

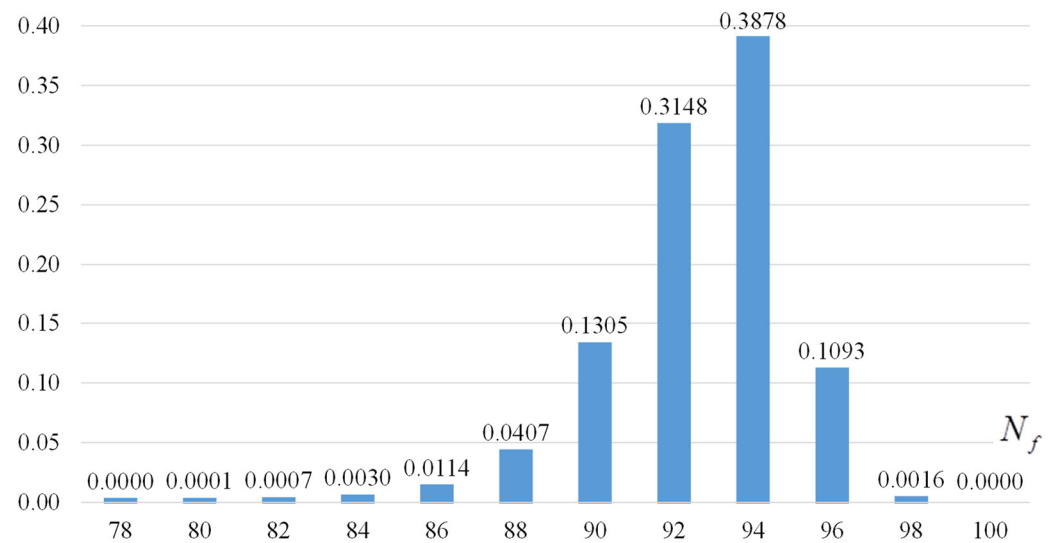


Figure 2. Histogram of nonlinearity values achieved during genetic algorithm execution (75,000 measurements across 50 independent runs).

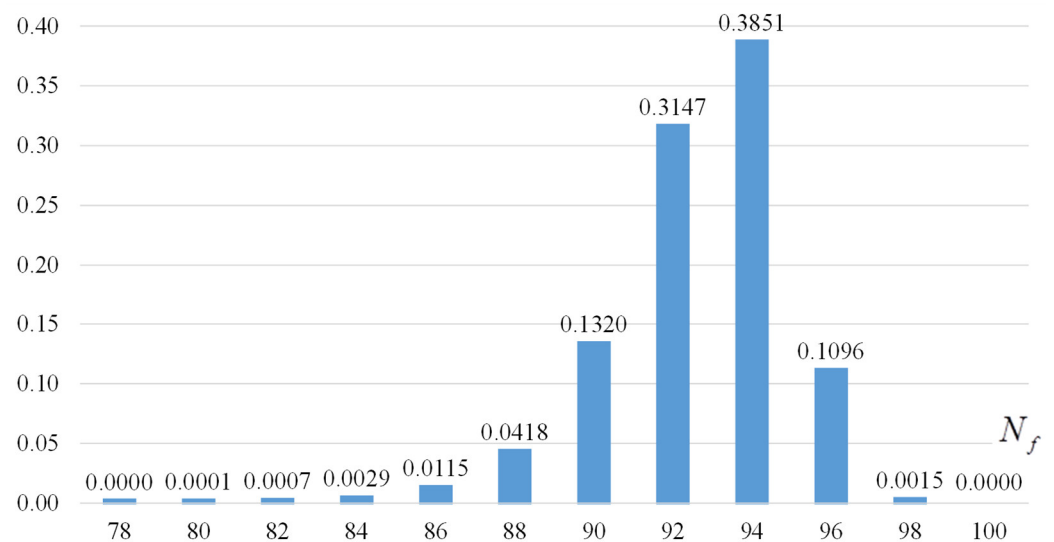


Figure 3. Distribution of nonlinearity values for randomly generated S-boxes (10^8 samples).

5.3. Parameter Optimization Analysis

To systematically evaluate HPHC performance, we conducted extensive parameter sweeps across the following:

- Population size: $K_{pop} \in [1, 21]$ with step 2;
- Mutation rate: $K_{mut} \in [1, 31]$ with step 3;
- 100 independent runs per configuration.

5.3.1. Evaluation Efficiency Analysis

Table 1 gives an extensive comparison for the average number of S-box evaluations required for different population sizes and different mutation rates to show the influence of parameters on the efficiency of algorithms.

Key findings from evaluation analysis (Table 1):

1. Optimal efficiency ($K_{Sbox} = 49,277$) achieved with the following:
 - Minimum population size ($K_{pop} = 1$);
 - Moderate mutation rate ($K_{mut} = 7$).

2. Performance degradation with larger populations:
 - Linear increase in evaluations with K_{pop} ;
 - Up to $4\times$ more evaluations at $K_{pop} = 21$.
3. Robust performance across mutation rates:
 - Effective range: $K_{mut} \in [4, 9]$;
 - Minimal variation in evaluation count.

Table 1. Average number of S-box evaluations (K_{Sbox}) required to achieve $N_f = 104$.

K_{mut}	Population Size (K_{pop})				
	1	5	9	13	21
1	58,213	72,830	101,726	112,718	149,339
7	49,277	77,848	103,154	122,382	165,918
13	56,042	83,216	101,073	124,466	171,756
19	56,532	82,883	105,144	129,718	175,531
25	50,066	79,596	115,462	135,294	183,087
31	53,709	91,827	109,625	143,233	192,493

5.3.2. Execution Time Analysis

For clarity on the practical implications of performance for different combinations of parameters, Table 2 summarizes the typical execution times to find S-boxes of a nonlinearity of 104 for different configurations.

Table 2. Average execution time (seconds) to find S-box with $N_f = 104$.

K_{mut}	Population Size (K_{pop})				
	1	5	9	13	21
1	64	17	15	16	42
7	57	21	9	13	28
13	44	26	8	13	22
19	56	24	8	14	19
25	46	18	16	14	18
31	69	20	14	15	18

Analysis reveals (Table 2):

1. Optimal execution time (8–9 s) with the following:
 - Population size matching thread count ($K_{pop} = 9$);
 - Mutation rate $K_{mut} \in [13, 16]$.
2. Thread utilization effects:
 - Suboptimal for $K_{pop} < N_{threads}$;
 - Diminishing returns for $K_{pop} > N_{threads}$.

5.3.3. Statistical Reliability Analysis

For detailed analysis, we performed additional experiments with $K_{pop} = [1, 2]$ and $K_{mut} \in [1, 30]$ (step 1). With the purpose of providing deeper insight into algorithm behavior in the case of utilizing few populations, Figure 4 shows the relation of mutation rates to evaluation counts with small population sizes, hence revealing important patterns about algorithm efficiency.

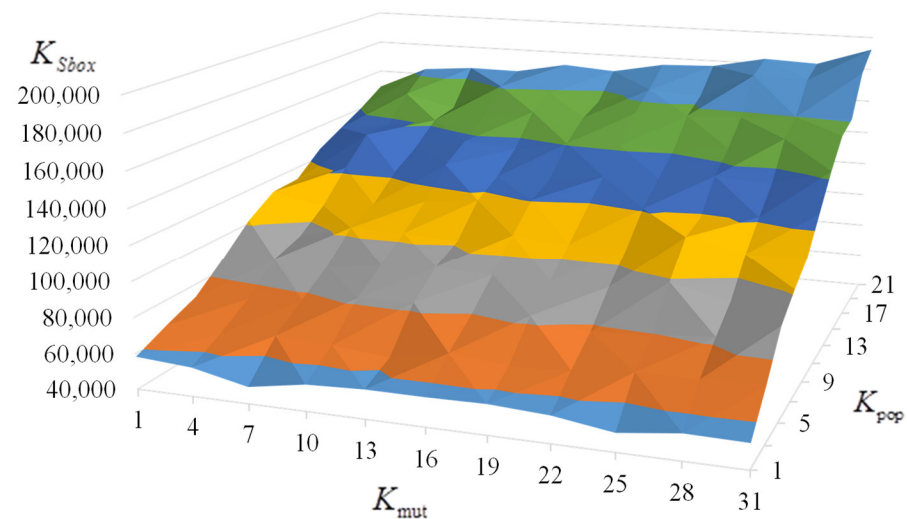


Figure 4. Average number of S-box evaluations required for $N_f = 104$ with small population sizes.

Key statistical findings (Figure 4):

1. Success rate:
 - 99% overall success rate;
 - 2% failure rate only with $K_{pop} = 1, K_{mut} = 1$;
 - 1% failures with $K_{pop} = 1, K_{mut} \in \{6, 8, 11\}$.
2. Variance analysis:
 - Standard deviation: 22,500 for $K_{pop} = 1$;
 - Increased variance (44,500) for $K_{pop} = 21$.

5.4. Parallel Performance Analysis

The parallelization scales well for a variety of hardware platforms. Using the test environment setup based on the multi-core CPU running at 3.49 GHz, optimal performances for population size $K_{pop} = 9$ and thread count $N_{threads} = 8$ achieve execution times of 8–9 s (Figure 5).

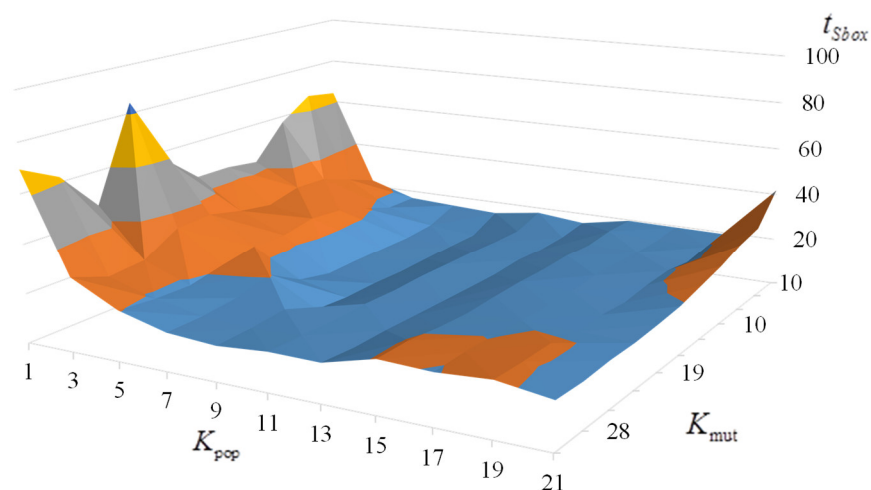


Figure 5. Average execution time versus population size for different mutation rates.

The parallel speedup shows an almost linear scaling up to the number of available cores. For two cores, one obtains $S_2 = 1.95$, corresponding to $E_2 = 0.97$. Using four cores, one has $S_4 = 3.82$ with $E_4 = 0.95$; whereas for eight cores, $S_8 = 7.44$ gives $E_8 = 0.93$. These results hint at very effective parallelization because the overhead due to thread

synchronization, as well as the communication of data between successive iterations, is negligible.

These scaling characteristics show that the speedup is near-linear until thread saturation; optimal performance is achieved when population size approximates the available number of threads: $K_{pop} \approx N_{threads}$. For $K_{pop} < N_{threads}$, performance becomes suboptimal because of underutilization of the available cores. For $K_{pop} > N_{threads}$, we see diminished returns because of increased synchronization overhead.

We conducted additional, more fine-grained experiments focusing on small population sizes, investigating, in detail, the region of optimum performance detected in the exploratory experiments. The analysis for $K_{pop} \in [1, 2]$ with mutation rates $K_{mut} \in [1, 30]$ (step 1) over 100 independent runs showed that single-individual populations reach average evaluations of about 53,000 for optimal mutation rates between 4 and 9, exhibiting a very consistent performance, with low variance in the number of evaluations (Figure 6).

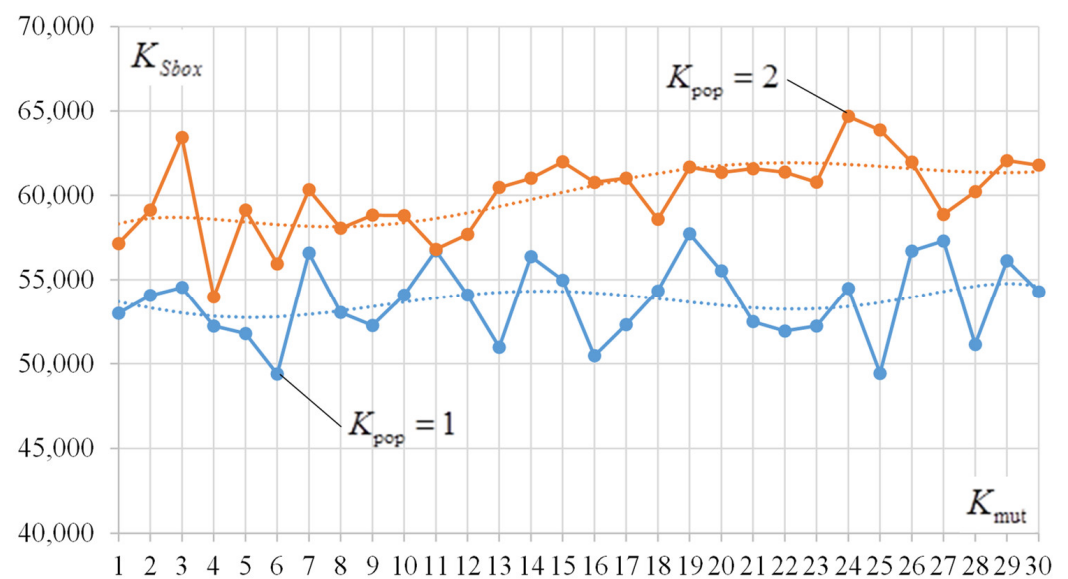


Figure 6. Average number of S-box evaluations (K_{Sbox}) required to achieve $N_f = 104$ for population sizes $K_{pop} = 1$ and $K_{pop} = 2$. The dotted lines show trend lines.

On the other hand, the populations of size two came back with somewhat higher evaluation counts yet proved much more stable in terms of mutation rates. Therefore, though increasing the computational overhead, performance became much more robust. Comparing these, we can confirm that $K_{pop} = 1$ gains the best overall efficiency in terms of raw evaluation count, which slightly weakens for larger mutation rates. The success rate for both configurations remains at 99%, clearly trading off between computational efficiency and stability.

5.5. Cost Function Evolution

Figure 7 shows the evolution of the WHS cost function values over some runs (highlighted in different colors) of each approach, allowing for an overview of its optimization dynamics, convergence pattern, and characteristics concerning the search landscape.

Analysis of the cost function trajectory reveals (Figure 7):

- Consistent convergence patterns;
- Strong correlation with a nonlinearity improvement;
- Smooth optimization landscape.

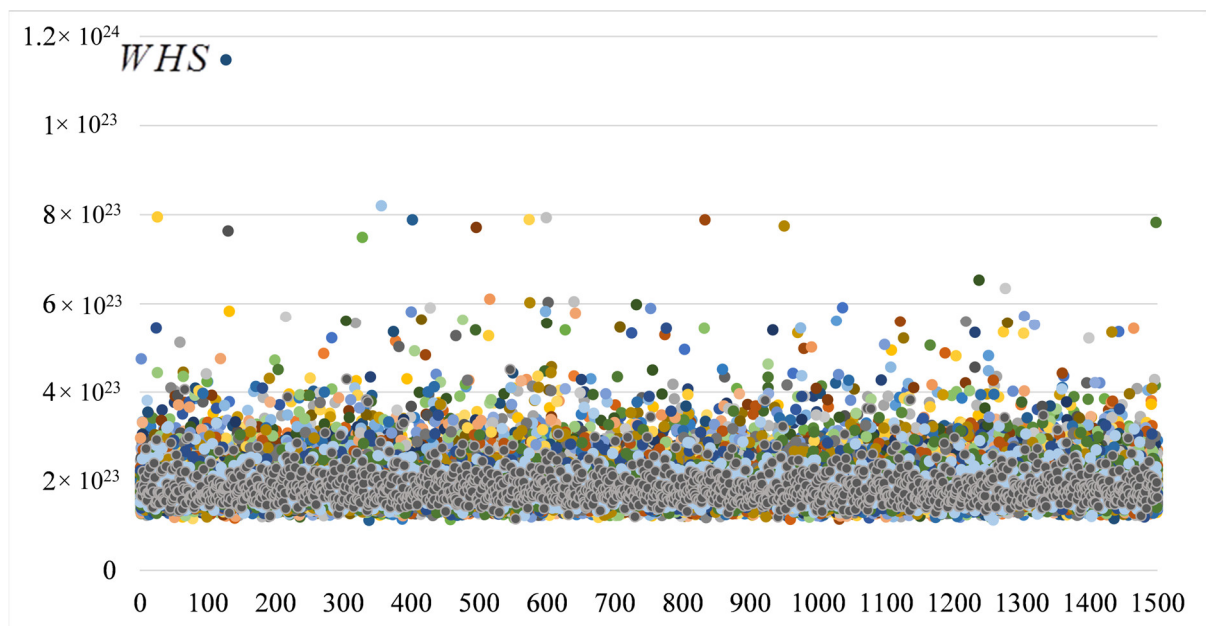


Figure 7. Evolution of WHS cost function values across multiple runs.

5.6. Comparative Analysis

Table 3 gives a detailed comparison between HPHC and the state-of-the-art methods to position our work within the current state of the art, as well as to highlight the benefits that might validate the efficiency and success rate enhancement.

Table 3. Comparison with state-of-the-art methods.

Method and Reference	Cost Function	Algorithm	Nonlinearity	Success Rate	Average Iterations
Clark et al. (2005) [7]	WHS	SA	102	1/200	-
McLaughlin (2012) [19]	WHS	SA	104	-	30,000,000
Tesar (2010) [20]	WHS	GaT	104	-	3,239,000
Picek et al. (2016) [22]	PCF	Ga	104	-	741,371
Picek et al. (2016) [22]	PCF	LSA	104	-	172,280
Picek et al. (2016) [22]	PCF	GaT	104	-	167,451
Freyre-Echevarría et al. (2020) [10,23]	WCF	LSA	104	-	89,460
Freyre-Echevarría et al. (2020) [10,23]	WCF	HC	104	11/30	65,933
Our HPHC	WHS	HPHC	104	99%	49,277

HPHC demonstrates the following significant advantages (Table 3):

- Efficiency improvements:
 - 600× faster than original SA methods;
 - 15× faster than original SA GA-based approaches;
 - Lowest reported evaluation count (49,277).
- Reliability metrics:
 - Highest success rate (99%);
 - Consistent performance;
 - Robust convergence behavior.
- Practical benefits:
 - Fast execution (8..9 s);
 - Scalable parallel implementation;
 - Resource-efficient operation.

These results establish HPHC as a significant advancement in S-box generation, combining theoretical efficiency with practical usability.

6. Discussion

The development and analysis of the HPHC algorithm provide significant insights into both theoretical and practical aspects of cryptographically strong S-box generation. Our extensive experimental results demonstrate that this hybrid approach successfully combines the advantages of hill climbing and population-based methods while mitigating their respective limitations.

The most striking finding is that smaller populations, particularly single-individual configurations, outperform traditional larger population approaches. This contradicts conventional wisdom in evolutionary computation but aligns with the specific nature of the S-box generation problem. When combined with an appropriate mutation rate ($K_{mut} = 7$), this minimal population approach achieves remarkable efficiency, requiring only 49,277 evaluations on average to generate an S-box with a nonlinearity of 104.

The parallel implementation strategy reveals another counterintuitive insight: while minimal populations provide the best sequential performance, configurations with population size matching the available thread count ($K_{pop} \approx N_{threads}$) achieve optimal real-world execution times. This demonstrates the importance of considering hardware architecture in algorithm design, as theoretical efficiency does not always translate directly to practical performance.

Our success rate of 99% significantly exceeds previous approaches, with failures occurring only in specific parameter configurations. This reliability, combined with execution times of 8–9 s on modern hardware, makes HPHC practical for real-world applications. The algorithm's performance represents a substantial improvement over existing methods, with a $600\times$ reduction in iterations compared to traditional simulated annealing and $15\times$ improvement over recent genetic algorithm variants.

Although our experiments reported considerable advances in generating S-boxes, some important limitations and concerns need to be discussed. Similarly, the performance of the algorithm is also dependent on hardware, especially the number of CPU cores and memory bandwidth. For such scenarios, even though it can be greatly optimized by a number of parameters, it introduces extra complexity in deploying them across different computing environments. The future work can investigate hardware-agnostic implementation or devise adaptive adjustment of parameters.

One of the other challenges is the theoretical insights regarding hybrid techniques, such as HPHC. While in practice, we can observe significant improvements in performances, designed algorithm connections are only partially understood in regard to its connections to search space exploration. This fact finds particular reflection in our ability to prove rigorous theoretical bounds with regard to convergence time or solution quality. Closing the gap between empirical success and mathematical justification will call for more theoretical research in the future.

While very important in practice due to their widespread usage in modern cryptographic schemes, the focus on 8-bit S-boxes constitutes a limitation regarding the generalizability of our approach. Preliminary experiments with other S-box sizes indicate that the basic principles of HPHC carry over to this setting, though the optimum parameter settings, as well as the performance characteristics, are likely different. Study of the applicability of HPHC to different sizes and types of S-boxes is thus of interest to the cryptographic community.

The other crucial issue is the selection of the cost function. Although, in light of our goals, the WHS function behaves really well, one cannot avoid fitting to this particular metric. Usually, in cryptographic applications, there is a set of various aspects that need to be optimized via S-boxes. It seems quite unlikely that all of them are captured by the optimization for one cost function. Other directions involve various techniques of

multi-objective optimization, or alternative cost functions reflecting better the set of all desired properties.

7. Conclusions

This paper has introduced the hybrid population-based hill-climbing (HPHC) algorithm for generating cryptographically strong S-boxes. Our approach achieves state-of-the-art performance in both theoretical efficiency and practical execution time, consistently generating S-boxes with a nonlinearity of 104 in under 50,000 evaluations.

The key contributions of this work include the following:

- A novel hybrid algorithm combining hill climbing efficiency with population-based exploration;
- Demonstration that minimal populations can outperform traditional evolutionary approaches;
- Efficient parallel implementation achieving 8–9 s execution times;
- Comprehensive empirical analysis of parameter effects and performance characteristics.

Our findings challenge several conventional assumptions about evolutionary computation and demonstrate the value of problem-specific algorithm design. The HPHC algorithm achieves a 99% success rate while requiring significantly fewer computational resources than existing methods, making it particularly suitable for practical applications in cryptographic system design.

Author Contributions: Conceptualization, writing—review and editing, O.K.; investigation, methodology, N.P.; writing—original draft preparation, K.K.; data curation, funding acquisition, E.F.; formal analysis, supervision, M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This project has received funding from the European Union’s Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No. 101007820—TRUST. This publication reflects only the author’s view, and the REA is not responsible for any use that may be made of the information it contains. This research was funded by the European Union—NextGenerationEU under the Italian Ministry of University and Research (MIUR), National Innovation Ecosystem Grant ECS00000041-VITALITY-CUP D83C22000710005.

Data Availability Statement: The original contributions presented in the study are included in the article. The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest: The authors declare no conflicts 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. Vijayakumar, M.; Ahilan, A. An Optimized Chaotic S-Box for Real-Time Image Encryption Scheme Based on 4-Dimensional Memristive Hyperchaotic Map. *Ain Shams Eng. J.* **2024**, *15*, 102620. [\[CrossRef\]](#)
3. Ji, Y.; Ma, Z.; Li, L.; Sun, Y. Constructions of Correlation Immune S-Boxes with High Nonlinearity. *J. Inf. Intell.* **2024**, *2*, 253–260. [\[CrossRef\]](#)
4. Freyre Echevarría, A. Evolución Híbrida de S-Cajas No Lineales Resistentes a Ataques de Potencia. Bachelor’s Thesis, La Habana, Cuba, 2020.
5. Álvarez-Cubero, J. Vector Boolean Functions: Applications in Symmetric Cryptography. Ph.D. Thesis, Universidad Politécnica de Madrid, Madrid, Spain, 2015.
6. Cusick, T.; Stănică, P. *Cryptographic Boolean Functions and Applications*, 2nd ed.; Academic Press: Cambridge, MA, USA, 2017; p. 275.
7. Clark, J.A.; Jacob, J.L.; Stepney, S. The Design of S-Boxes by Simulated Annealing. *New Gener. Comput.* **2005**, *23*, 219–231. [\[CrossRef\]](#)
8. Kapuściński, T.; Nowicki, R.K.; Napoli, C. Application of Genetic Algorithms in the Construction of Invertible Substitution Boxes. In Proceedings of the Artificial Intelligence and Soft Computing, Zakopane, Poland, 12–16 June 2016; Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 380–391.

9. Ivanov, G.; Nikolov, N.; Nikova, S. Reversed Genetic Algorithms for Generation of Bijective S-Boxes with Good Cryptographic Properties. *Cryptogr. Commun.* **2016**, *8*, 247–276. [CrossRef]
10. 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]
11. Ivanov, G.; Nikolov, N.; Nikova, S. Cryptographically Strong S-Boxes Generated by Modified Immune Algorithm. In Proceedings of the Cryptography and Information Security in the Balkans, Istanbul, Turkey, 16 October 2014; Pasalic, E., Knudsen, L.R., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 31–42.
12. Carlet, C. Vectorial Boolean Functions for Cryptography. In *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*; Cambridge, University Press: Cambridge, UK, 2006.
13. Nyberg, K. Perfect Nonlinear S-Boxes. In Proceedings of the Advances in Cryptology—EUROCRYPT '91, Brighton, UK, 8–11 April 1991; Davies, D.W., Ed.; Springer: Berlin/Heidelberg, Germany, 1991; pp. 378–386.
14. 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.
15. Millan, W. How to Improve the Nonlinearity of Bijective S-Boxes. In *Proceedings of the Information Security and Privacy*; Boyd, C., Dawson, E., Eds.; Springer: Berlin/Heidelberg, Germany, 1998; pp. 181–192.
16. Burnett, L.D. Heuristic Optimization of Boolean Functions and Substitution Boxes for Cryptography. Ph.D. Thesis, Queensland University of Technology, Brisbane, Australia, 2005.
17. Courtois, N.T.; Bard, G.V. Algebraic Cryptanalysis of the Data Encryption Standard. In Proceedings of the Cryptography and Coding, Cirencester, UK, 18–20 December 2023; Galbraith, S.D., Ed.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 152–169.
18. Bard, G.V. *Algebraic Cryptanalysis*; Springer US: Boston, MA, USA, 2009; ISBN 978-0-387-88756-2.
19. McLaughlin, J. Applications of Search Techniques to Cryptanalysis and the Construction of Cipher Components. Ph.D. Thesis, University of York, York, UK, 2012.
20. Tesar, P. A New Method for Generating High Non-Linearity S-Boxes. *Radioengineering* **2010**, *19*, 23–26.
21. Ahmad, M.; Khaja, I.A.; Baz, A.; Alhakami, H.; Alhakami, W. Particle Swarm Optimization Based Highly Nonlinear Substitution-Boxes Generation for Security Applications. *IEEE Access* **2020**, *8*, 116132–116147. [CrossRef]
22. Picek, S.; Cupic, M.; Rotim, L. A New Cost Function for Evolution of S-Boxes. *Evol. Comput.* **2016**, *24*, 695–718. [CrossRef] [PubMed]
23. Freyre Echevarría, A.; Martínez Díaz, I. A New Cost Function to Improve Nonlinearity of Bijective S-Boxes. *Symmetry* **2020**, *12*, 1896. Available online: https://www.researchgate.net/publication/343699912_A_new_cost_function_to_improve_nonlinearity_of_bijective_S-boxes (accessed on 30 November 2024).
24. Kuznetsov, A.; Poluyanenko, N.; Frontoni, E.; Kandiy, S.; Peliukh, O. A New Cost Function for Heuristic Search of Nonlinear Substitutions. *Expert Syst. Appl.* **2024**, *237*, 121684. [CrossRef]
25. 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]
26. Kuznetsov, A.; Poluyanenko, N.; Frontoni, E.; Kandiy, S.; Pieshkova, O. Optimized Simulated Annealing for Efficient Generation of Highly Nonlinear S-Boxes. *Soft Comput.* **2023**, *28*, 3905–3920. [CrossRef]
27. Kuznetsov, O.; Poluyanenko, N.; Frontoni, E.; Kandiy, S. Enhancing Smart Communication Security: A Novel Cost Function for Efficient S-Box Generation in Symmetric Key Cryptography. *Cryptography* **2024**, *8*, 17. [CrossRef]
28. 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.
29. 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]
30. 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 and T 2021–Proceedings, Kharkiv, Ukraine, 5–7 October 2021; pp. 434–438.
31. 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 and T 2021–Proceedings, Kharkiv, Ukraine, 5–7 October 2021; pp. 458–464.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.