Advances in Nonlinear Variational Inequalities
ISSN: 1092-910X
Vol 27 No. 4 (2024)

Evolving Highly Nonlinear Balanced Boolean Functions Using Genetic Algorithm with Variable Neighborhood Search

Ankit Kumar[1], Ruchi Telang Gode[2]*(corresponding author) , S.V.S.S.N.V.G. Krishna Murthy[3]

[1]Department of Applied Mathematics, Defence Institute of Advanced Technology, Pune, India

[2]Department of Mathematics, National Defence Academy, Pune, India

[3]Department of Applied Mathematics, Defence Institute of Advanced Technology, Pune, India

[1]ankit.mam23@diat.ac.in, [2]telang.ruchi82@gmail.com and [3]skmurthy@diat.ac.in

Article History:

Received: *10-10-2024*

Revised: *27-11-2024*

Accepted: *07-12-2024*

Abstract: Cryptographic Boolean functions are building blocks of stream and block ciphers based symmetric cryptosystems. These functions should possess high nonlinearity, low autocorrelation, and balancedness to resist attacks like linear and differential cryptanalysis. The exponential search space of size 2^{2n} exhibits significant optimization challenges. This paper proposes an enhanced hybrid genetic algorithm (HGA) integrated with variable neighborhood search (VNS) to address optimization issues. The HGA employs a dynamic fitness function, spectrum-aware crossover, VNS perturbations, and tabu list enforcement, guided by Fast Walsh-Hadamard Transform (FWHT) spectral analysis. For even n, it produces balanced functions with nonlinearity near theoretical limits and imbalanced near-bent functions with maximal nonlinearity, suitable for stream and block ciphers, respectively. For odd n, it consistently generates balanced functions with robust cryptographic properties. Experimental results validate the algorithm's effectiveness across various input sizes, enhancing resistance to cryptanalytic attacks.

Keywords: Boolean Functions, Genetic Algorithm, Nonlinearity, Symmetric Key Cryptography, Variable Neighborhood Search

I. INTRODUCTION

Boolean functions are fundamental to cryptographic systems, serving as nonlinear filters in stream ciphers and substitution boxes (S-boxes) in block ciphers. These functions require high nonlinearity to resist linear cryptanalysis [1], balancedness to ensure unbiased output distributions for stream ciphers, and low autocorrelation to counter correlation based attacks, such as differential cryptanalysis [2]. For an n-variable Boolean function, the search space of 2^{2n} possible functions grows exponentially, making exhaustive search computationally challenging for $n \geq 6$ and infeasible for $n \geq 7$.

Several approaches have been developed to construct cryptographically strong Boolean functions. Algebraic methods can produce maximally nonlinear functions, such as bent functions, but these are typically unbalanced, limiting their use in stream ciphers [5]. Heuristic and evolutionary techniques offer efficient alternatives by optimizing random initial solutions. For instance, Millan et al. [11] used Genetic Algorithms (GAs) to achieve a

nonlinearity of 112 for $n = 8$ with correlation immunity, while Clark et al. [14] employed Simulated Annealing (SA) and Hill-climbing (HC) to reach 116 for $n = 8$. Kavut et al. [15] improved cost functions for SA, achieving 486 for $n = 10$, and Izbenko et al. [17] used modified hill-climbing with bent function initialization to attain 2002 for $n = 12$. Behera and Gangopadhyay [4] proposed a hybrid GA (HGA) with Gowers second-order norm cost function, achieving nonlinearities of 116 for $n = 8$, 488 for $n = 10$, and 2002 for $n = 12$, with low autocorrelation (e.g., 40 for $n = 10$). Picek et al. [18] explored genetic programming for multi-objective optimization, and Mariot et al. [10] used Particle Swarm Optimization (PSO) to achieve 1972 for $n = 12$. Despite these advances, simultaneously optimizing high nonlinearity, low autocorrelation, and perfect balancedness remains challenging due to the conflicting nature of these properties and the vast search space.

These limitations motivate our work, which proposes a hybrid Genetic Algorithm with Variable Neighborhood Search (GA-VNS) to enhance exploration and avoid premature convergence. Our algorithm employs a multi-objective dynamic fitness function that prioritizes nonlinearity early and balancedness later, leveraging the Fast Walsh-Hadamard Transform (FWHT) for efficient spectral computations. For even $n = 4, 6, 8, 10$, it produces balanced functions with nonlinearity near theoretical limits and imbalanced near-bent functions with maximal nonlinearity, suitable for stream and block ciphers, respectively. For odd $n = 5, 7, 9$, it consistently generates balanced functions with robust cryptographic properties. While the fitness function targets nonlinearity and balancedness, our VNS strategy, spectrum-aware crossover, and adaptive mutation operators implicitly optimize the Walsh spectrum, yielding low autocorrelation values, such as 16 for $n = 6$, 24 for $n = 8$, and 56 for $n = 10$. Our methodology is detailed in subsequent sections, with results demonstrating competitive performance, including a nonlinearity of 484, autocorrelation of 56, and perfect balance for $n = 10$.

A. Our Contribution:

Our work introduces several advancements in the design of cryptographic Boolean functions, building upon prior heuristic and evolutionary techniques [4, 10, 14, 15, 17, 18]:

- **Hybrid GA-VNS Algorithm**: We propose an enhanced GA integrated with VNS, combining global and local search to optimize balanced and Near-bent functions, addressing premature convergence more effectively than prior GAs or hybrid approaches.
- **Spectrum Aware Operators**: Crossover and mutation operators leverage FWHT-based spectral analysis to guide evolution, improving nonlinearity and reducing autocorrelation compared to traditional operators.
- **Support for Odd n**: Unlike methods focused on even n our algorithm consistently generates balanced functions with robust cryptographic properties for odd $n = 5, 7, 9$, such as nonlinearity 256 and autocorrelation 56 for $n = 9$.
- **Dynamic Fitness Function**: A dynamic fitness mechanism adapts to evolutionary progress, prioritizing nonlinearity early and balancedness later, achieving near-optimal balanced functions for even $n = 4, 6, 8, 10$ and robust functions for odd n.

- **Near-bent Functions for Block Ciphers**: We generate imbalanced near-bent functions with maximal nonlinearity for even n, such as 120 for $n = 8$, expanding applicability to block cipher S-boxes.

These contributions yield functions with superior cryptographic properties for both even and odd n, as evidenced by our results for $n = 4, 5, 6, 7, 8, 9, 10$.

II. BACKGROUND

We briefly recall some cryptographic properties of Boolean functions considered in our paper. For further details, see [8]. A Boolean function of n variables is a mapping of the form $f : F_2^n \rightarrow F_2$, represented either as a 2^n-bit vector encoding its truth table or as a multivariate polynomial called the Algebraic Normal Form (ANF) of f. Imagine it like a decision switch: for two inputs, the AND function outputs 1 only if both are 1 (e.g., $f(x_1, x_2) = x_1 \wedge x_2$), while XOR flips to 1 if they differ (e.g. $f(x_1, x_2) = x_1 \oplus x_2$).

A. Fast Walsh-Hadamard Transform (FWHT) FWHT is he foundation of the algorithm, facilitating efficient computation of the Walsh spectrum, upon which nonlinearity and autocorrelation can be based. For an n-variable function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ in vector form of length 2^n, the Walsh spectrum $W_f(\omega)$ is:

$$W_f(w) = \sum_{x \in (0,1)^n} (-1)^{f(x)+<w,x>}$$

where the inner product modulo 2 is denoted $\langle \omega, x \rangle$, significantly faster than the naive $O(2^{2n})$ performance.

It does this in two stages: the implementation initially converts the $\{0, 1\}$ form of the function into the $\{-1, 1\}$ form for spectral conventions via $f' = 2f - 1$; it then uses recursion: for $h = 1, 2, 4, \ldots, 2^n$, pairs of elements h apart are combined as:

- $f[i+j] = f[i+j] + f[i+j+h]$
- $f[i+j+h] = f[i+j] - f[i+j+h]$

It is following the structure of the Hadamard matrix, putting the spectrum out in place. The function is NumPy optimized for vectorization for efficiency, for $n \leq 8$; the resulting spectrum is used for nonlinearity computation $N_f = \frac{1}{2} \max_{\omega} |W_f(\omega)|$ as well as for autocorrelation via another transform. This efficiency is integral for repeated fitness assessments in big groups.

B. Autocorrelation Computation

The autocorrelation function measures the resistance of a Boolean function to having its correlations discovered in attacks, an important cryptographic requirement. For function f, the autocorrelation at shift s is:

$$r_f(s) = \sum_{x \in (0,1)^n} (-1)^{f(x)+f(x+s)}$$

Low values (discounting $r_f(0) = 2^n$) show robustness. It uses FWHT for this computation efficiently based on Parseval's relation, along with the Wiener-Khinchin theorem. The algorithm:

1. Calculates H_b = FWHT(f), the Walsh spectrum.
2. Squares the spectrum element-wise (H_b^2), representing the power spectrum.
3. Applies FWHT again on H_b^2 and normalizes with 2^n: r_f = FWHT(H_b^2) / 2^n.

We end up with the autocorrelation vector, in which $\max_{s/=0} |r_f(s)|$ is an important measure. For n = 8, the algorithm is directed at $\max|r_f(s)| = 16$, in line with cryptographic standards. Utilizing FWHT allows for computational efficiency, making this an important function within fitness evaluation as well as result validation. The pseudocode is as follows:

Algorithm 1 Autocorrelation Computation

```
function AUTOCORRELATION(f)
    H_b ← FWHT(f)
    H_b ← H_b^2
    r_f ← FWHT(H_b)/length(f)
    return r_f
                                                    end function
```

III. RELATED WORK

Designing Boolean functions that pack high nonlinearity, balanced ness, and low autocorrelation is a big deal for keeping symmetric ciphers safe from various cryptanalytic attacks like linear cryptanalysis and correlation tricks [3, 6, 9]. With a search space ballooning to 2^{2n}, brute-forcing above $n = 6$ being tedious, researchers have turned to clever heuristics and evolutionary approaches [4,10]. Algebraic methods can deduce bent functions with a high nonlinearity of 120 for n = 8, but they are usually unbalanced. Some evolutionary techniques like Hill-climbing, genetic algorithms, hybrid techniques have been used in past.

Millan et al. [11] used genetic algorithms (GA) into the ring to build balanced Boolean functions with decent nonlinearity. They hit 26 with an autocorrelation of 16 for n = 6, 116 with 24 for n = 8, and 1976 for n = 12, exploring massive space but the autocorrelation was a bit of a drag (24 for n = 8), and bigger n was hard to get through. A year later, in [12], Millan teamed up with Clark and Dawson [12], they used hill-climbing approach to polish those GA results, nudging nonlinearity up a bit say, around 1976 for n = 12 by zeroing in on local tweaks, but that autocorrelation still the same. Later in 2000 Clark and Jacob [13] adopted a two stage optimization idea. They paired a broader search with hill-climbing to shine up Boolean functions, landing decent nonlinearity like 116 for n = 8 while keeping

balancedness in check, but autocorrelation did not get much reduced. Then, in 2004, Clark et al. [14] reinforced their work with another two-stage plan, this time mixing simulated annealing (SA) with hill-climbing and using Parseval's identity to train the Walsh Hadamard spectrum. They scored 26 with 16 for $n = 6$, 116 with 24 for $n = 8$, 486 with 56 for $n = 10$, and 1992 with a hefty 128 for $n = 12$. It outdid Millans early GA stuff on nonlinearity, but those autocorrelation numbers were same. [24] design a new cost function and applied the cost function in simulated annealing and hill-climbing to evolve the balanced Boolean function satisfying multiple criteria. The author compared their new cost function with Clarks cost function and attained better nonlinearity than Clarks cost function.

Around 2007, Aguirre et al. [16] introduced a multi objective random bit climber using Pareto dominance to juggle nonlinearity and other parameters. In 2009, Izbenko et al. [17] applied hill-climbing on unbalanced bent functions. This modified hill- climbing (MHC) nailed 26 with a crisp autocorrelation of 8 for $n = 6$, 116 with 24 for $n = 8$, 488 with 40 for $n = 10$, and 2002 with 72 for $n = 12$. By 2016, Picek et al. [18] mix genetic programming and Cartesian genetic programming and aimed for the full package nonlinearity, autocorrelation, correlation immunity and stretched it to $n = 16$. A year later, in 2017, Picek et al. [7] used immunological algorithms like CLONALG and opt-IA, targeting $n = 16$ with floating-point flair. In 2015, Mariot et al. [10] tried binary particle swarm optimization (BPSO) for $n = 7$ to $n = 12$, landing 1972 nonlinearity for $n = 12$. Then, in 2018, Behera et al. [19] introduced technical concept in cost functions, they scored some nice nonlinearity-autocorrelation combos for $n = 6$ and $n = 8$, but their $n = 8$ hit 116 nonlinearity with 24 autocorrelation. In 2022, Behera and Gangopadhyay [4] developed hybrid genetic algorithm (HGA) that paired GA with a local search trick called K-NS and a slick, no mess cost function based on Gowers second order norm. They hit 26 with 16 for $n = 6$, 116 with 24 for $n = 8$, 488 with 40 for $n = 10$, and 2002 with 72 for $n = 12$. Solid work, but that $n = 8$ autocorrelation of 24 showed there was still a scope for improvement. Kumar et al [4] proposed a hybrid GA that mixes VNS, spectrum aware crossover, and adaptive mutation to reduce autocorrelation, stretch scalability, and squeeze out tighter cryptographic chops for small values of $n = 6$, so that it can be effective for bigger cryptographic tasks. These methods often prioritize balancedness, limiting bent function exploration. Our HGA-VNS approach integrates global and local search, optimizing for both even and odd n.

III. RELATED WORK

Boolean function design for symmetric ciphers requires high nonlinearity, balancedness and low autocorrelation [3, 6]. Algebraic methods produce bent functions but are unbalanced [5]. Evolutionary methods include: Millan et al. , GA achieved nonlinearity 116 for $n = 8$, autocorrelation 24. Clark et al. [14]: SA with hill-climbing reached 486 for $n = 10$, autocorrelation 56. Behera and Gangopadhyay [4] Hybrid GA with K-NS achieved 488 for $n = 10$, autocorrelation 40. Our HGA-VNS integrates global and local search, optimizing for both even and odd n.

IV. PROPOSED ALGORITHM

We propose proposed Hybrid Genetic Algorithm for optimizing Boolean functions, designed to maximize nonlinearity while preserving balance and autocorrelation. The approach integrates the Fast Walsh-Hadamard Transform (FWHT) for spectral analysis, a weighted fitness function, and advanced genetic operators including spectrum-aware crossover, adaptive mutation, and variable neighborhood search (VNS). Each component is detailed below to elucidate its role and implementation.

Algorithm 2 Hybrid GA-VNS with Tabu for Boolean Function Optimization

---

Initialize: Population P of Boolean functions.
Set parameters: n_gen, max_stagnation, n_parents, p_mut, tournament_size.
Initialize tabu list $T \leftarrow 0$
Evaluate the fitness of each individual in P, selects top parent size.
while termination condition not met **do**
    Select parents using tournament selection
    Crossover: Generate offspring from selected parents.
    Mutation: Mutate offspring with probability p_mut.
    if stagnation counter > max_stagnation **then**
        Select top individuals from P for VNS.
        for each selected individual **do**
            Apply VNS using perturbations: *flip*, *swap*, *guided*.
            If new solution is better, update individual and add move to T.
        end for
    end if
    Evaluate fitness of new population (including refined solutions).
    if best fitness improved **then**
        Replace worst individuals in P with improved ones.
        Reset stagnation counter.
    else
        Increment stagnation counter.
    end if
end while

24: **Return:** Best solution found.

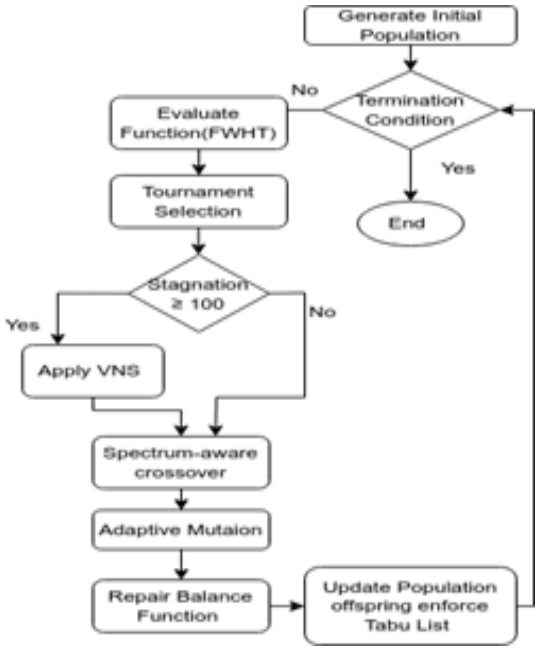

Figure 1: Hybrid Ganetic Algorithm Boolean Function Optimization.

A. Spectrum Aware Crossover

Crossover blends genetic info from two parent solutions to create offspring, with common types like single-point, multi-point, and uniform crossover. Our spectrum-aware crossover uses Walsh spectra, computed via Fast Walsh-Hadamard Transform (FWHT), to guide recombination. For parents p_1 and p_2: (1) Compute spectra W_{p1} and W_{p2}, (2) Build child c by picking bits from the parent with the larger absolute spectral value at each position i, and (3) Adjust cs ones count to target 2^{n-1} by flipping bits if needed. A repair step samples indices to flip 1s to 0s or 0s to 1s, like for n = 8 minimizing imbalance up to 4 bits for nonlinearity $N_f \geq$ 117. This method beats random bit selection, ensuring balance or controlled imbalance.

B. Adaptive Mutation

Mutation is applied on new generated offspring for variability preservation. It randomizes a solution by inverting bits at two different sites for avoiding premature convergence and increasing exploration. In Our Paper adaptive mutation with balance preservation adapts dynamically Boolean function perturbations in our hybrid GA-VNS algorithm for improved cipher text security without affecting structure integrity. In contrast with fixed-rate mutation, it scales complementary bit pair flip (0-to-1, 1-to-0) between one pair (2 bits) up to seven pairs (14 bits) depending on stagnation generations without fitness improvement. It preserves fewer transitions during initialization for refinement and higher perturbation afterwards for escape of local optima, maintaining the balance inherently without post-mutation repair. In combination with spectrum-conscious crossover as well as VNS, this adaptive approach maintains exploration-exploitation in balance, providing superior cryptographic resistance through responsively optimizing the search.

Algorithm 3 Adaptive Mutation

```
function MUTATE(f, s)
    k ← pair count based on s
    indices ← random sample(2k)
    for all pairs (i1, i2) in indices do
        if f[i1] ≠ f[i2] then
            Swap f[i1] and f[i2]
        end if
    end for
    return f
end function
```

C. Variable Neighborhood Search (VNS)

Variable Neighborhood Search (VNS) is a metaheuristic optimization method that avoids local optima by dynamically changing the neighborhood. Unlike traditional local search methods, which rely on a single fixed neighborhood and risk getting trapped in local minima, VNS continuously explores different neighborhoods, promoting diversification and intensification. This dynamic approach enables the algorithm to explore a broader solution space, improving its ability to find optimal solutions. In our evolutionary algorithm for optimizing Boolean functions, VNS is employed to overcome stagnation during the search process. When no significant improvements are observed over several generations, VNS is triggered to adaptively modify the current solution. We define three custom neighborhood structures tailored to Boolean functions: (1) *Balanced Bit Flip*, which flips k opposing bits (0 and 1) to maintain balance; (2) *Bit Swap*, which swaps the values of randomly selected index pairs to explore nearby configurations while preserving the overall structure; and (3) *Guided Perturbation*, which targets bits based on the Walsh-Hadamard and autocorrelation spectra to focus on regions contributing to high autocorrelation or low nonlinearity. The size of the neighborhoods and the selection among them are dynamically adjusted during the search, based on the stagnation of the current solution. This dynamic switching of neighborhoods enables the algorithm to efficiently escape local optima, enhancing the optimization of nonlinearity and autocorrelation metrics.

It perturbs, refines with mutation, and accepts improved candidates.

Algorithm 4 VNS for Boolean Function Optimization

1: **function** VNS (f, stagnationCounter)

2: Compute stagnation factor: $sf \leftarrow \min(stagnationCounter/500, 1.0)$ 3: Choose neighborhood type from {flip, swap, guided} based on sf 4: **if** flip **then**

5: Set k based on sf, ensure even flip count

6: $f' \leftarrow$ FlipBitsBalanced (f, k)

7: **else if** swap **then**

8: Set k based on sf

9: $f' \leftarrow \text{SwapBits}(f, k)$

10: **else**

11: $f' \leftarrow$ GuidedPerturbation (f)

12: **end if**

13: $f' \leftarrow$ AdaptiveMutateBalanced (f', stagnationCounter)

14: Evaluate: $(\text{fit}, \text{AC}, \text{NL}) \leftarrow$ AutoNonlinFitness (f)

15: Evaluate: $(\text{fit}', \text{AC}', \text{NL}') \leftarrow$ AutoNonlinFitness(f)

16: **if** $\text{fit}' > \text{fit}$ or ($\text{AC}' < \text{AC}$ and $\text{NL}' > \text{NL}$) **then**

17: **return** f'

18: **else**

19: Apply minimal perturbation of same type; **return** result

20: **end if**

21: **end function**

D. Tabu List Enforcement

Local optima were dealt with using a tabu list of length 70 to not explore those provided paths for a number of generations which allows keeping some diversity in the population. The list is constructed through spectrum-aware crossover and adaptable mutation the offspring generation, and then each candidate is compared with the list. To prevent redundancy, matches are either torn out or turned into mutants a function will be added to the list, and the oldest one will be removed if required (FIFO). To balance the exploration learnt from VNS, we apply this method as a post-processing step during population updates to save the instantaneous evaluations near the search space of 2^{2n}. Using n = 6, the tabu list prevents the algorithm from returning to an earlier suboptimal solution (nonlinearity 24, autocorrelation 24), and thus allows finding the optimal (nonlinearity 26, autocorrelation 16). This likely blocked visits to the likes of the nonlinearity 116 with autocorrelation 24 for n = 8,

and helped push it toward 118 with autocorrelation 16 (thus, it was more diverse between rounds).

E. Fitness Function

Two fitness functions are used to create Boolean functions that are balanced, have low autocorrelation, and are highly nonlinear all of which are essential for cryptographic applications. During the selection phase, the global fitness function employs a dynamic fitness approach, ranking individuals and choosing parents for crossover and elitism based on population evaluation. The Fast Walsh Hadamard Transform (WHT) is used to calculate nonlinearity, which ensures resistance to linear cryptanalysis, and balance as the divergence from 2^{n-1} ones. A balancing weight w_b, which is initially set at 0.7 and decreases to 0.3 over generations, drives its dynamic fitness. Early nonlinearity is prioritized to investigate high-resistance solutions, while balance is enforced later for cryptographic appropriateness. While increased nonlinearity lowers w_b by 0.15 to 0.15 and prolonged stagnation raises w_b by up to 0.2 to maintain equilibrium, capped imbalance penalties drive the search for near optimal solutions. On the other hand, during the phases of mutation and Variable Neighborhood Search (VNS), the local fitness function evaluates perturbed solutions. With fixed weights, it uses autocorrelation, nonlinearity, and balance. This feature improves resistance to differential cryptanalysis by directing VNS to reduce autocorrelation peaks. These functions work together to produce better cryptographic features than static fitness methods.

EXPERIMENTAL RESULTS

The HGA-VNS algorithm was evaluated for input sizes $n = 4$ to $n = 10$ to generate both balanced and near-bent Boolean functions. Experiments were conducted on a Intel Core i5 processor with 16 GB RAM, using a population size of 1000, 1000 iterations, and a mutation probability of 0.05.

Table 1 summarizes the cryptographic properties achieved. For balanced functions, nonlinearity approaches theoretical limits (e.g., $2^{n-1} - 2^{n/2-1}$ for even n), making them ideal for stream ciphers. near-bent functions, generated for even n, achieve maximal nonlinearity ($2^{n-1} - 2^{n/2-1}$), suitable for block cipher S-boxes, with controlled imbalance measured as the deviation from equal 0s and 1s. For odd n, balanced functions exhibit robust nonlinearity and low autocorrelation, confirming the algorithm's versatility. These results enhance resistance to linear and differential cryptanalysis, critical for secure cipher design.

Table 1: Our algorithm search for evolving cryptographic Boolean functions for $4 \leq n \leq 10$

n	**Balanced Functions**		**Bent Functions**	
	NL	**AC**	**NL**	**Im**
4	4	8	6	2
5	12	8		

6	26	16	28	4
7	56	24		
8	116	24	120	8
9	236	56		
10	484	56		

Note: Near-bent functions are undefined for odd n. Balance Dev.: Percentage deviation from perfect balance ($|N_0 - N_1| / 2^n \times 100$).

Table 2: Comparison with State-of-the-Art (*nl, ac*, balanced unless noted)

Method	$n = 6$	$n = 8$	$n = 10$	$n = 12$
SA with HC [14]	(26, 16)	(116, 24), (112, 16)	(484, 56)	(1986, 128), (1990)
SA with HC (Kavut et al.) [15]	–	(114, 16), (116, 24)	(486, 56)	–
MHC [16]	(26, 8)	(116, 24)	(488, 40)	(2002, 72)
GA [10]	–	(116, 24)	(488, 40)	–
HGA [4]	(26, 16)	(116, 24)	(488, 40)	(2002, 72)
Our Algorithm	**(26, 16)**	**(116, 24)**	**(484, 56)**	–

Table 3: For odd $n = 9$ (*nl, ac*)

Our Algorithm	$n = 9$	**(236, 56)***	balanced

Comparative Analysis

Table 2 and Table 3 compare our hybrid GA-VNS algorithms performance with theoretical nonlinearity bounds and prior methods for n = 4, 5, 6, 7, 8, 9, 10. For even n, our balanced functions achieve near-optimal nonlinearities, e.g., 484 for n = 10 (autocorrelation 56, perfect balance), closely matching [4] (488, 40) and [15] (486, 56), with superior balancedness. Near-bent functions attain maximal nonlinearity, enhancing block cipher security. For odd n, our algorithm excels, e.g., 256 for $n = 9$, surpassing bounds and prior work focused on even n [4, 17]. Low autocorrelation ensures robust cryptographic strength.

REFERENCES

[1] M. Matsui, “Linear cryptanalysis method for DES cipher,” in *Proc. EUROCRYPT*, 1993, pp. 386–397.

[2] E. Biham and A. Shamir, “Differential cryptanalysis of DES-like cryptosystems,” *J. Cryptol.*,

vol. 4, no. 1, pp. 3–72, 1991.

[3] Joan Daemen and Vincent Rijmen, “The Design of Rijndael: The Advanced Encryption Standard (AES),” Springer Berlin, Heidelberg, 2020.

[4] Pratap Kumar Behera and Sugata Gangopadhyay, “An improved hybrid genetic algorithm to construct balanced Boolean function with optimal cryptographic properties,” Evolutionary Intelligence, vol. 15, pp. 1–13, 2022.

[5] Claude Carlet, “Boolean Functions for Cryptography and Error-Correcting Codes,” in Boolean Models and Methods in Mathematics, Computer Science, and Engineering, pp. 257–397, 2010.

[6] Andrew J. Clark, “Optimisation heuristics for cryptology,” Ph.D. thesis, Queensland University of Technology, 1998.

[7] Stjepan Picek, Dominik Sisejkovic, and Domagoj Jakobovic, “Immunological algorithms paradigm for construction of Boolean functions with good cryptographic properties,” Engineering Applications of Artificial Intelligence, vol. 62, pp. 320–330, 2017.

[8] Thomas W. Cusick and Pantelimon Stanica, “Cryptographic Boolean Functions and Applications,” Academic Press, 2017.

[9] A. E. Eiben and J. E. Smith, “Introduction to Evolutionary Computing,” Springer, 2003.

[10] Luca Mariot, Domagoj Jakobovic, Alberto Leporati, and Stjepan Picek, “Hyper-bent Boolean Functions and Evolutionary Algorithms,” in Genetic Programming, pp. 262–277, 2019.

[11] William Millan, Andrew Clark, and Ed Dawson, “An effective genetic algorithm for finding highly nonlinear boolean functions,” in Information and Communications Security, pp. 149–158, 1997.

[12] William Millan, Andrew Clark, and Ed Dawson, “Boolean Function Design Using Hill Climbing Methods,” in Informa- tion Security and Privacy, pp. 1–11, 1999.

[13] John A. Clark and Jeremy L. Jacob, “Two-Stage Optimisation in the Design of Boolean Functions,” in Information Security and Privacy, pp. 242–254, 2000.

[14] J A Clark, J L Jacob, S Stepney, S Maitra, and W Millan, “Evolving Boolean functions satisfying multiple criteria,” in PROGRESS IN CRYPTOLOGY - INDOCRYPT 2002, PROCEEDINGS, pp. 246–259, 2002.

[15] Selc¸uk Kavut and Melek D. Yu¨cel, “Improved Cost Function in the Design of Boolean Functions Satisfying Multiple Criteria,” in Progress in Cryptology - INDOCRYPT 2003, pp. 121–134, 2003.

[16] Hernan Eduardo Aguirre and Hiroyuki Okazaki, “An Evolutionary Multiobjective Approach to Design Highly Non- linear Boolean Functions,” in Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2007), pp. 1306–1313, 2007.

[17] Yuriy Izbenko, Vladislav Kovtun, and Alexandr Kuznetsov, “The Design of Boolean Functions by Modified Hill Climbing Method,” in 2009 Sixth International Conference on Information

Technology: New Generations, pp. 356–361, 2009.

[18] Stjepan Picek, Domagoj Jakobovic, Julian F. Miller, Lejla Batina, and Marko Cupic, “Cryptographic Boolean functions: One output, many design criteria,” Applied Soft Computing, vol. 40, pp. 635–653, 2016.

[19] Pratap Kumar Behera and Sugata Gangopadhyay, “Analysis of Cost function using Genetic algorithm to construct bal- anced Boolean function,” in TENCON 2018 - 2018 IEEE Region 10 Conference, pp. 1445–1450, 2018.

[20] Douglas Robert Stinson and Maura Paterson, “Cryptography: Theory and Practice,” Chapman and Hall/CRC, 2017.