IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 26 October 2022, accepted 12 November 2022, date of publication 28 November 2022,
date of current version 7 December 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3224924

RESEARCH ARTICLE

Test Scheduling and Test Time Minimization of System-on-Chip Using Modified BAT Algorithm

GOKUL CHANDRASEKARAN[1], NEELAM SANJEEV KUMAR[2], KARTHIKEYAN P. R.[3], VANCHINATHAN K.[1], NEERAJ PRIYADARSHI[4], (Senior Member, IEEE), AND BHEKISIPHO TWALA[5], (Senior Member, IEEE)

[1]Department of Electrical and Electronics Engineering, Velalar College of Engineering and Technology, Erode 638012, India
[2]Department of Biomedical Engineering, Saveetha School of Engineering, SIMATS, Chennai 602105, India
[3]Department of Electronics and Communication Engineering, Saveetha School of Engineering, SIMATS, Chennai 602105, India
[4]Department of Electrical Engineering, JIS College of Engineering, Kolkata 741235, India
[5]Digital Transformation Portfolio, Tshwane University of Technology, Pretoria 0183, South Africa

Corresponding author: Bhekisipho Twala (twalab@tut.ac.za)

ABSTRACT System-on-Chip (SoC) is a structure in which semiconductor components are integrated into a single die. As a result, testing time should be reduced to achieve a low cost for each chip. Effective test scheduling can reduce the SoC testing time, which is more challenging due to its complexity. In this paper, the modified BAT algorithm-based test scheduling is proposed. Testing is carried out on the SoC ITC'02 benchmark circuits. The Modified Bat method is a recently heuristic algorithm that performs global optimization by imitating bat echolocation. Compared to other state-of-the-art algorithms, the Modified BAT Optimization method reduces testing time on SoCs. This paper improves the algorithm's exploration process by adjusting the equation for bat loudness (A_0) and pulse emission rate (r). The modified BAT algorithm converges to the optimal solution faster. It has been used in 14 international standard test functions. The test results indicate that the modified BAT algorithm has a fast convergence speed, which minimizes the testing time compared to other evolutionary algorithms on the ITC'02 SoC benchmark circuits.

INDEX TERMS Test time, system-on-chip, modified BAT algorithm, BAT algorithm, test scheduling.

I. INTRODUCTION

According to Moore's law, the integration level doubles every 1.5 years because of the improvement in semiconductor technology. Complex functional electronic products at the micro-level are developed due to this advancement in IC design. The SoC contains over a million gates, memory, cores, and analog and digital blocks on a single chip. On a larger scale, IC design and manufacturing present significant challenges such as power consumption and thermal issues. Because testing requires a lot of switching, the test mode requires more thought than the standard model. In SoC testing, modular testing is an effective strategy for embedded cores [1], [2].

The transmission of test data to cores is a significant problem at the system level. The test wrappers isolate the cores by adapting the Test Access Mechanism (TAM) width to the inputs-outputs of wrappers. Figure 1 shows the SoC testing block diagram. The wrapper and TAM optimization are critical since test time varies with wrappers and TAM width design. It also depends on how TAM wires are attached to the wrapper chains. CPU, memory, DSP, and mixed-signal modules are some of the cores that are used. To reduce test time, multiple cores are tested concurrently. All requirements must be approachedwith caution. The major constraint in the core-based design is the interface between TAM and cores. Test responses and stimuli are stored and analyzed [3].

There are different description levels for hardware in which cores can be categorized and used as hard, soft, and firm cores. In soft cores, more implementation is performed by the designer. They are independent of the process and also flexible. The hard cores are developed for expectable performance, but are not flexible. The firm core characteristics lie between soft and hardcore. Each core is tested as a separate

The associate editor coordinating the review of this manuscript and approving it for publication was Zhenzhou Tang.

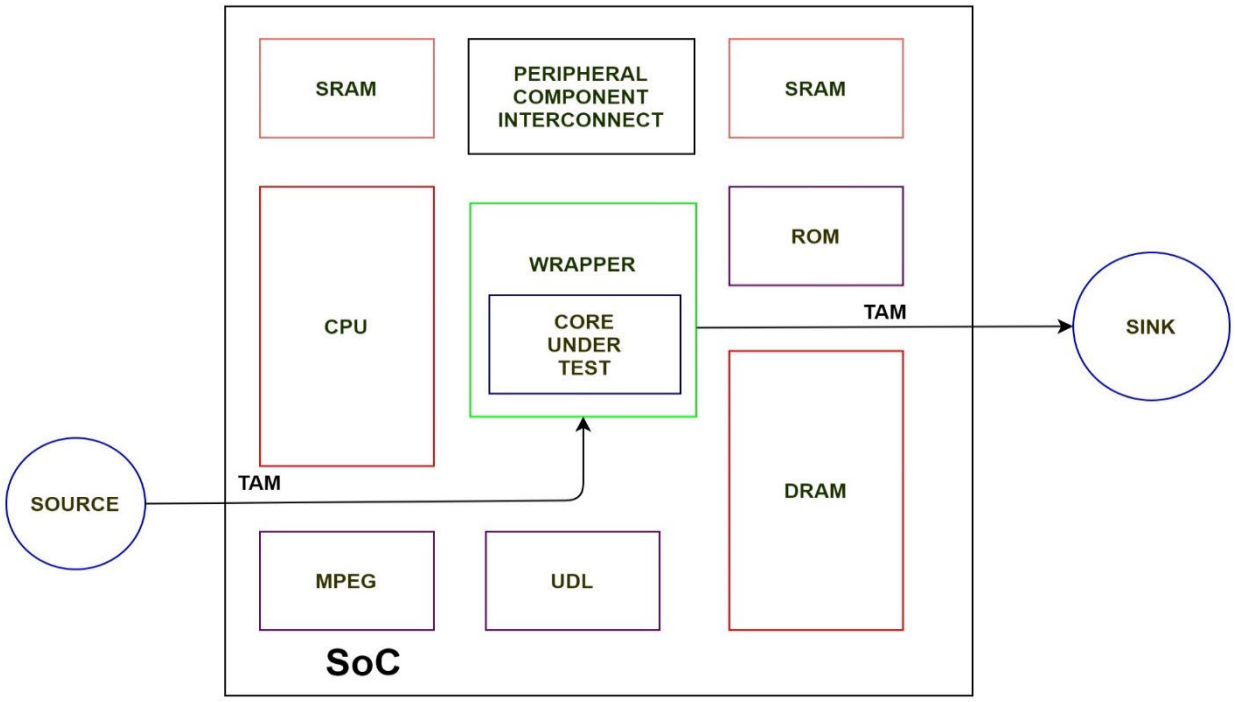

FIGURE 1. System-on-chip architecture.

unit in the modular core-based design. The system integrator located the cores on the die to test with their test data. These cores cannot be accessed directly from the SoC pins. Test resources and cores are accessed through the design of the test architecture. So,the test schedule is the major challenge of the modular test approach.

In test scheduling, the main operation is accessing the testing order of data where the testing of each core consumes less time. Cores are arranged in various ways in a test schedule called an NP-hard problem. While performing test scheduling, the maximum value must not be reached for the core's power and TAM width. There are three scheduling techniques for the core: partitioned, non-partitioned, and pre-emptive techniques for scheduling. The core is immediately scheduled in partitioned scheduling after testing the previous core. No new core will be tested in a non-partitioned schedule till all the core tests are carried out during the session. Core testing may be interrupted in a pre-emptive technique and resumed later, but all the cores must be tested. The major challenge leading to considerable test time in the scheduling process is the occurrence of test data volume in an enormous amount. Simultaneous core testing in a modular fashion is done to overcome this challenge. The cores are depicted in Figure 2. The bin region that has not been filled is referred to as idle time.

The benchmark circuits are a collection of benchmarks proposed at the International IEEE Test Conference [4]. Table 1 details the ITC'02 benchmarks. Tables 2 and 3 provide details of the D695 and P22810 benchmarks.

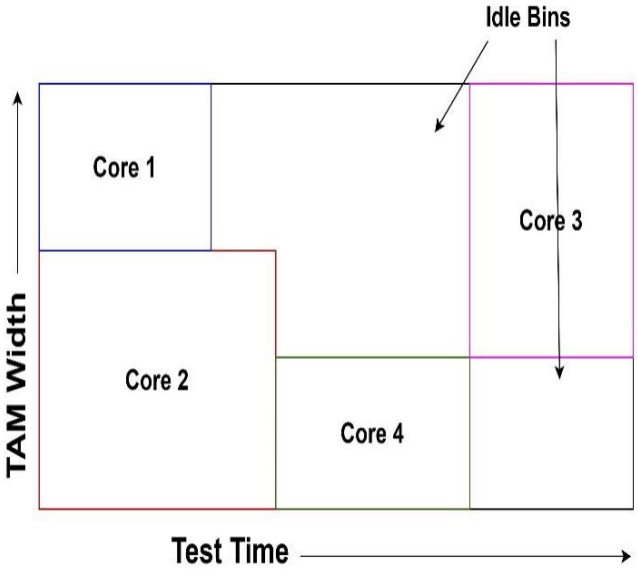

FIGURE 2. SoC test scheduling.

Under various load conditions of the SoC benchmark circuits, the role of the optimization techniques in testing is to reduce the testing time taken as the objective function shown in equation (1) below.

$$T(W_i) = (1 + \max(S_i, S_o)).t_{pi} + \min(S_i, S_o) \quad (1)$$

TABLE 1. Details of benchmarks.

SOC		D695	P22810
No. of Tests		10	30
No. of Levels		2	3
No. of Modules		11	29
No. of SFFs		6384	24723
No. of I/O's		1845	4283
No. of Scan Chain Length		137	196
Pattern Count	Maximum	234	12324
	Average	88	830
	Minimum	12	1
Scan Chain Length	Maximum	55	400
	Average	46	126
	Minimum	32	1

where S_o and S_i are the scan chain length of the output and input, and t_{pi} is the core i test pattern for the benchmark circuit for optimization.

This research suggests optimal test scheduling algorithms for reducing test time. The proposed strategy can lower test costs by indirectly reducing test time.

The contributions to the paper are as follows:

1. The Modified BAT algorithm is used for the test schedule to achieve the optimal testing time of the System-on-Chip.

2. The efficiency of heuristic algorithms is evaluated by computing their performance over a range of TAM widths on the two ITC'02 benchmarks, D695 and P22810.

The proposed Modified BAT algorithm is compared to seven state-of-the-art algorithms for SoC test scheduling. Based on the results from MATLAB simulations, the proposed algorithm is evaluated under various TAM widths, reducing the test time than the other heuristic algorithms.

II. MOTIVATION

Test application time increases with an increase in the complexity of the system. This paper, uses various optimization approaches such as GOA, MFO, MVO, CPSOGA, BBO, SCA, and Modified BAT algorithms to optimize the test time for d695 and p22810 circuits. The given cost function is minimized by solving the SoC test planning problem to minimize the test time. The motivation of this research is to reduce testing time analysis for various TAM widths.

III. LITERATURE REVIEW

Darwin's evolutionary algorithms, which rely on the survival of the most important principle, could be considered optimization techniques. According to Darwin's theory, the fittest individuals are more likely to survive in nature [5]. Genetic Algorithms (GA) work on solution representation in binary format, whereas evolution strategies and differential evolution support solutions with absolute values [6]. The GA implementation methodology is of two levels. The initial solution is provided at the first level for GA. This method can improve efficiency and throughput in the second level with only a tiny amount of increased design space—PSO-based non-preemptive test scheduling proposed by Xu et al. [7]. Ant Colony algorithm has been applied to many optimization areas. ABC performs better in large-scale global, numerical, and combinational optimization.

The BAT algorithm improves outcomes for optimizing lower-dimensional problems, whereas it is difficult for higher-dimensional problems since the BAT algorithm converges faster. The differential evolution algorithm uses selection, crossover, and mutation to apply to continuous function optimization. Genetic Algorithm (GA), Harmony Search Method (HSM), ACO, Simulated Annealing (SA), and PSO are meta-heuristic approaches applied to resolve optimization problems identified by researchers. The algorithms described above deal with continuous and discrete variables that do not need any objective function gradient and handle both continuous and discrete variables. The issue of SoC hierarchy is discussed in the hierarchical awareness of the test planning system used for TAM optimization. Table 4 shows various recent heuristic optimization algorithms. In Ant Colony Optimization (ACO) [8], [9], at each iteration, when the test core number reduces, the test time increases, leading to an ineffective output, but the count remains constant. This problem is solved by the Modified ACO technique [10]. The Modified ACO examines the ant count will decrease if the core count decreases.

The Grasshopper Optimization Algorithm (GOA) [11] algorithm's optimization process is inspired by the movement of a grasshopper colony. Their actions resemble the solution to the optimization problem. The Moth Flame Optimization (MFO) [12] is the simulation of moths' unique ability to navigate at night. The moth navigated at night using a process known as transverse orientation. The Multi-Verse Optimizer (MVO) algorithm [13] is based on three cosmological ideas: wormholes, white holes, and black holes. Exploitation and exploration are carried out with the help of mathematical models of these three concepts. In the CPSOGSA [14], [15] algorithm, the diversification strength of GSA is combined with the high exploitation capabilities of PSO. The ability of the CPSOGSA to avoid local minima stagnation and accelerate convergence will be tested.

TABLE 2. D695 SoC benchmark details.

Core	Internal Scan Chains	Number of Tests Patterns	Number of inputs	Max Chain Length	Number of Outputs	Min Chain Length
1	0	12	32	0	32	0
2	0	73	207	0	108	0
3	1	75	34	32	11	32
4	4	105	36	54	39	52
5	32	110	38	45	304	44
6	16	234	62	41	152	39
7	16	95	77	34	150	33
8	4	97	35	46	48	44
9	32	12	35	54	320	54
10	32	68	28	55	106	51

The ecological distribution of organisms inspires the BBO algorithm [16] in a specific ecosystem. The Sine Cosine Algorithm (SCA) [17] generates several fluctuating and random solutions. SCA has a fast convergence speed, which helps to avoid local optima.

Rohini et al. [39] proposed DVFS-based SoC test scheduling solutions for optimization through selecting clock frequency and supply voltages election. Marrouche et al. [40] proposed the Process Algebra (PA) algorithm. To save time, Karaboga et al. [41] proposed the Fuzzy and PA algorithms. Test scheduling is concurrent, and various constraints are to be considered. In Strength Pareto Evolutionary Algorithm (SPEA) [42], the mutation and crossover operators are used every generation.

A solution is found by swarm-based optimization algorithms using the trial-and-error method. The most effective swarm-based optimization technique is the behavior of peer-to-peer learning. The recent count of this category is the ABC optimization algorithm. It contains a group of probable solutions similar to other optimization techniques dependent on population. The food source of a honey bee indicates potential solutions. The amount of nectar (food source quality) determines fitness. In modified ABC population's average fitness value helps to converge faster and prevents premature convergence [43]. The firefly algorithm simulates the natural behavior of fireflies.

The intensity of light or brightness is found using the objective problem function of the firefly's attractiveness β.

$$\beta = \beta_0 e^{-\gamma r^2} \tag{2}$$

Modified Firefly Algorithm decreases the randomness to reduce the probability [43], [44].

Test time is minimized by exchanging and compressing tests in one system. When the difficulty (i.e. the number of test alternatives) increases, the optimization time increases. The research presents a testing approach in SoC that combines test wrapper, test access design, test data compression, and testing schedule.

IV. PROPOSED WORK

A. BAT ALGORITHM

BAT algorithm is an approach focused on the behaviour of bat echolocation. Bats can discriminate the various types of insects and find their prey even in complete darkness. Bats employ a variety of sonar to sense echolocation techniques to prey. They trace their staying splits and evade difficulties in the dark. During prey search, bats fly with the velocity vi at the frequency f_{min}, changing their loudness A0 and wavelength λ. They emit pulse wavelength, which can be adjusted automatically. This pulse emission rate is dependent on the target proximity. Most bats employ echolocation only to a particular degree. Microbats use echolocation extensively, and this behavior makes them famous [45], [46]. Figure 3 represents the Testing Time vs. Input data's of D695 SoC. When micro bats release a loud sound pulse, they observe its echo, which bounces from the things around them. Based on the vibrations, species differ in properties associated with the hunting strategy. Depending on the type of species, the bandwidth of their signals varies and gets improved by added harmonics. Each pulse exists only up to 8 to 10ms, but it has a frequency of 20-150 kHz, which is constant. When the prey is hunted, the pulse emission rate speeds up to 200 pulses per second. The emitted pulse intensity is at a maximum of 110dB, which is in the ultrasonic

TABLE 3. P22810 SoC benchmark details.

Core	Internal Scan Chains	Number of Tests Patterns	Number of inputs	Max Chain Length	Number of Outputs	Min Chain Length
1	0	10	10	0	67	0
2	0	89	10	0	67	0
3	10	785	28	15	56	14
4	0	12324	47	0	33	0
5	0	3108	38	0	26	0
6	0	222	48	0	64	0
7	29	202	90	41	112	39
8	0	712	80	0	64	0
9	0	2632	84	0	64	0
10	0	2608	36	0	16	0
11	24	175	116	55	123	51
12	4	38	50	15	30	13
13	8	94	56	18	23	16
14	11	93	40	23	23	21
15	4	1	68	12	149	12
16	3	108	22	31	15	26
17	6	37	84	23	42	21
18	1	8	13	31	43	23
19	4	25	223	21	69	12
20	5	644	53	28	11	26
21	3	58	38	9	29	9
22	4	124	45	14	40	12
23	10	465	115	17	76	16
24	3	59	54	8	40	7
25	7	40	31	14	8	14
26	5	27	73	19	23	18
27	18	215	58	24	46	23
28	31	181	66	35	33	34
29	1	2	285	6	94	4
30	5	26	48	10	43	9

region. The intensity is high while the bat searches for its prey, whereas the intensity is lesser while coming towards the prey. Microbats avoid small obstacles like human hair. They build up a three-dimensional surroundings situation using time variance among two ears. They can sense the prey's moving speed, orientation, and target distance. The wing flap rate of target insects induces a Doppler effect that helps the bats discriminate the targets. These bats are susceptible to smell and also have good eyesight. Bats employ a mixture of all the senses for smooth navigation and effectual prey recognition.

Necessary steps in BAT algorithm:

STEP 1: Parameters are initiated by setting the original values.

TABLE 4. Potential applications of various algorithms P22810 SoC benchmark details.

Reference	Year	Algorithm	Performance Metrics
Wang et al. [18]	2020	Hybrid EHMO	Fitness values, Convergence analysis, Routing
Iwendi et al. [19]	2021	Hybrid WOA-SA	Statistical tests, Convergence analysis, Clustering
Masdari et al. [20]	2020	Modified TLBO	Fitness values, Clustering
Alghamdi [21]	2020	Hybrid DA-FF	Run time, Clustering
Iwendi et al. [19]	2021	ABO	Statistical tests, Convergence analysis, Routing
Mishra et al. [22]	2020	Hybrid LEACH-ACO	Fitness values, Routing
Dahiya et al. [23]	2020	Hybrid HAGOA	Statistical tests, Convergence analysis, Routing/Clustering
Gupta &Saha [24]	2020	Hybrid DEABC	Fitness values, Clustering
Chawra& Gupta [25]	2020	Modified Memtic	Run time, Convergence analysis, Clustering
Anurag et al. [26]	2020	Modified PSO	Statistical tests, Coverage
Yildiz, Pholdee, Bureerat, et al. [27]	2020	SCA	Convergence analysis
Panagant et al. [28]	2020	SOA	Convergence analysis
Yildiz [29]	2020	MBA	Exploitation curves, Fitness function values
Ozkaya et al.[30]	2020	EO	Convergence analysis
Patel et al. [31]	2021	GOA and others	Exploitation curves
Fouad et al. [32]	2019	CSO	Exploitation curve analysis, Objective function values
Sarangkum et al. [33]	2019	Multi-objective optimizers	Statistical measures
Li et al. [34]	2019	DE	Mean shift, convergence graphs, and Ncuts
Xu et al. [35]	2019	DA, DE	SSIM, Statistical test
Kandhway & Bhandari [36]	2019	WCA	PSNR, Threshold values
Resma & Nair [37]	2018	KHO	Convergence Curves, Threshold values
Yildiz& Yildiz [38]	2018	GWO& others	Convergence analysis

STEP 2: Generation of the random initial population where the initial position 'x_0', initial velocity 'v_0', and initial frequency 'f_i' is considered a solution.

STEP 3: For every solution, objective function is calculated, and the original population is evaluated. The values of r_i and A_i are initialized by fine-tuning the velocity 'v_i' and position 'x_i' so that the generation of the population is made.

$$f_i = f_{\min} + \beta(f_{\max} - f_{\min}) \tag{3}$$
$$v_i^t = v_i^{t-1} + f_i(x_i^{t-1} - x^*) \tag{4}$$
$$x_i^t = v_i^t + x_i^{t-1} \tag{5}$$

where β is a random number following a uniform distribution.

STEP 4: The target function is measured.

STEP 5: The local search iteration method is used every time. Therefore, the best-found solution can be improved.

STEP 6: A new solution is created randomly and accepted by a parameter with certain proximity. Whenever the volume of pulse emission increases, the loudness decreases. As given below, A_i and r_i values get updated.

$$A_i^{t+1} = A_i^t \alpha \tag{6}$$
$$r_i^t = r_i^0[1 - \exp(-\gamma\xi)] \tag{7}$$

where γ and α are constants.

STEP 7: The best solution is selected.

Repeat the process until the criteria for termination are satisfied, and the complete solution is obtained. The BAT algorithm uses the echolocation techniques of the bats where obstacles are detected and prevented with the help of sonar rays. The pulse rate varies from 0 to 1, where 0 refers to the maximum emission rate.

B. MODIFIED BAT ALGORITHM

The BAT algorithm's exploration process is enhanced by adjusting r and A [47], [48], [49], [50], [51], [52]. The

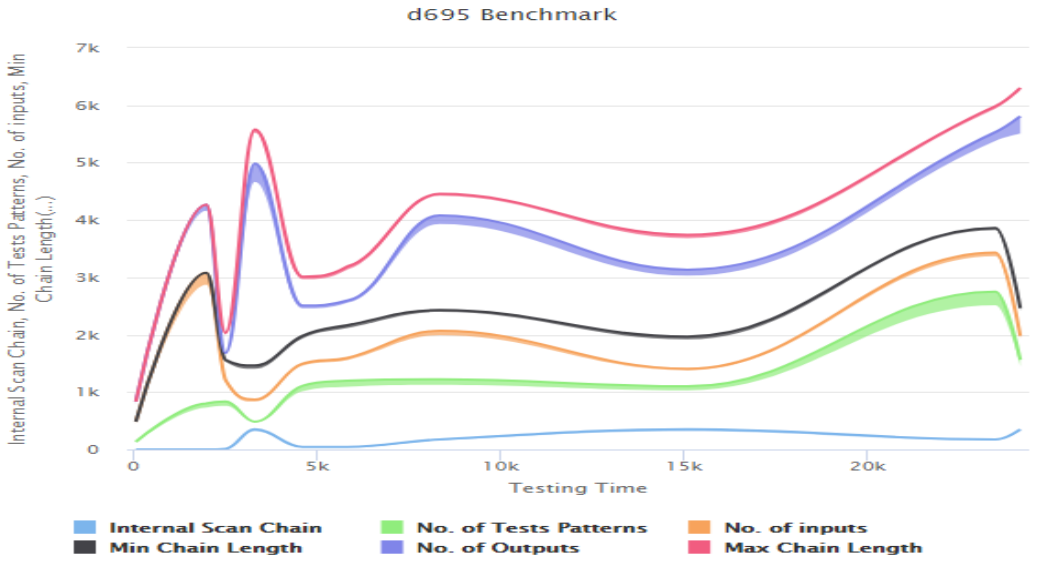

FIGURE 3. Testing time vs. input data's of D695 SoC.

proposed modified BAT algorithm explores search space more effectively by enhancing the exploration component. In the BAT algorithm, the bat searches the space and possesses only one loudness and pulse emission rate.

In the modified BAT algorithm based on the problem's dimension, pulse rate and loudness are balanced to an equal number. BAT can search the space effectively using this modified BAT approach. Echolocation is the mechanism employed by bats for hunting. Depending on echolocation capability, bats may locate their food and differentiate insects even in darkness. The Modified BAT algorithm can be applied for various benchmarks like multimodal, unimodal, and shifted types with different dimensions by considering upper and lower boundaries from the real-valued vectors with the number 'n' and the dimension 'd'.

$$x_{i,j} = rand(0, 1)(x_{\max j} - x_{\min j}) + x_{\min j} \tag{8}$$

where i = 1, 2 . . . n, and j = 1, 2 . . . d;

$x_{\max j}$, $x_{\min j}$ are the dimension jth upper and lower boundaries. The frequency, position, and velocity of BAT for future iterations are evaluated using the equations 3, 4 and 5.

$\beta \in [0, 1]$, x_* denotes the solution of BAT having the best fitness value. β represent a randomly generated number. Among the best solution, one of the existing solutions is selected, and hence a new candidate solution is made using the random walk.

$$x_{new} = x_{old} + \varepsilon \bar{A}^t \tag{9}$$

Pulse emission rate and loudness are initialized to generate the population by adjusting velocity and position given in the form of the equations 6 and 7.

V. RESULTS AND DISCUSSIONS

The Modified BAT algorithm is tested on 14 benchmark functions (f_1 to f_{14}) with varying dimensions, as shown in equations 10 to 23, at the bottom of the page 9. All test functions are minimization problems, and the data for this test function is taken from the literature. A test function is a tool for determining whether a global or local optimum exists. It is adequate for algorithm benchmarking exploration.

TABLE 5. Parameters initialization of all algorithms.

Algorithm	Name of the parameter
For all algorithms	Maximum Iterations = 100 Number of Dimensions = 100 Number of Particles count = 60
GOA	c_{Max}= 1, c_{Min}= 0.00004
MFO	Constants a = -1, b = 1
MVO	Probability of Wormhole Existence: WEP_Max = 1 (Maximum Value) WEP_Min = 0.2 (Minimum Value)
CPSOGA	Control Parametersφ_1,φ_2 = 2.05
BBO	KeepRate=0.2, $P_{mutation}$ = 0.9
SCA	Constant a = 2
Modified BAT	Loudness A = 0.25, Pulse rate r = 0.1, Maximum & Minimum frequency Q_{max}= 2, Q_{min}= 0

Figures 3 and 4 show the D695 SoC benchmark data. The input data are the internal scan chain, inputs and test patterns, minimum chain length, number of outputs, and maximum chain length.

Figure 4 represents a 3D model of the test function. The algorithm's performance is measured using Standard deviation, Mean and Best. A common approach for measuring the algorithm's effectiveness is the Mean Value Test (MV), which characterizes the algorithm's convergence accuracy and optimization capabilities by providing the data set's mean value. Standard Deviation (SD), which displays the SD of the data set and demonstrates the stability of the improved

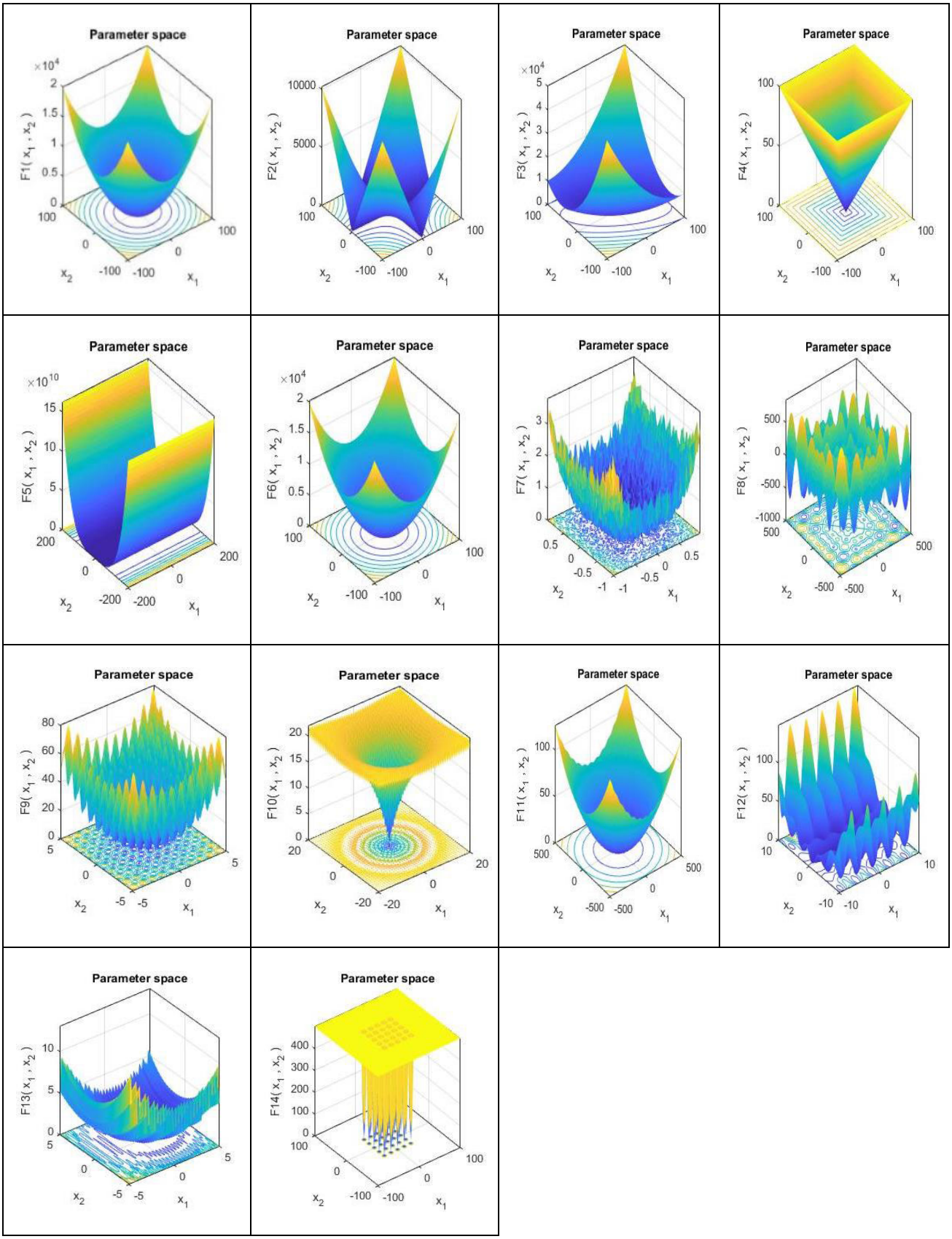

FIGURE 4. 3D model of 14 test functions.

algorithm, is another good technique for evaluating the algorithm's performance.

Before running the algorithms GOA, MFO, MVO, CPSOGA, BBO, SCA, and Modified BAT, several parameters must be set. Table 5 shows the initial values of the parameters for GOA, MFO, MVO, CPSOGA, BBO, SCA, and Modified BAT algorithms.

The seven algorithms use the same experimental parameters during the testing process: the particle is 60, the number of iterations is 100, and the F1-F14 dimensions are 100. A 500GB memory Intel(R) Core(TM) i3 - 7100T processor running at 3.40GHz powers the experimental test environment. The MATLAB version is MATLAB (2019b). The optimization results of the seven algorithms are compared in Table 6.

$$f_1 = \sum_{i=1}^{n} x_i^2 \tag{10}$$

$$f_2 = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i| \tag{11}$$

$$f_3 = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2 \tag{12}$$

$$f_4 = \max_i \{|x_i|, 1 \le i \le n\} \tag{13}$$

$$f_5 = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 - (x_i - 1)^2] \tag{14}$$

$$f_6 = \sum_{i=1}^{n} [(x_i + 0.5)^2] \tag{15}$$

$$f_7 = \sum_{i=1}^{n} i x_i^4 + random(0, 1) \tag{16}$$

$$f_8 = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{|x_i|}\right) \tag{17}$$

$$f_9 = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10] \tag{18}$$

$$f_{10} = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e \tag{19}$$

$$f_{11} = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \tag{20}$$

$$f_{12} = \frac{\pi}{n}\left\{10\sin(\pi y_i) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4) \tag{21}$$

$$y_i = 1 + \frac{x_i + 1}{4}$$

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m; & x_i > a \\ 0 \\ k(-x_i - a)^m; & x_i < -a \end{cases}$$

$$f_{13} = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4) \tag{22}$$

$$f_{14} = -\sum_{i=1}^{n}\sin(x_i).\left(\sin\left(\frac{i x_i^2}{\pi}\right)\right)^{2m}, \quad m = 10 \tag{23}$$

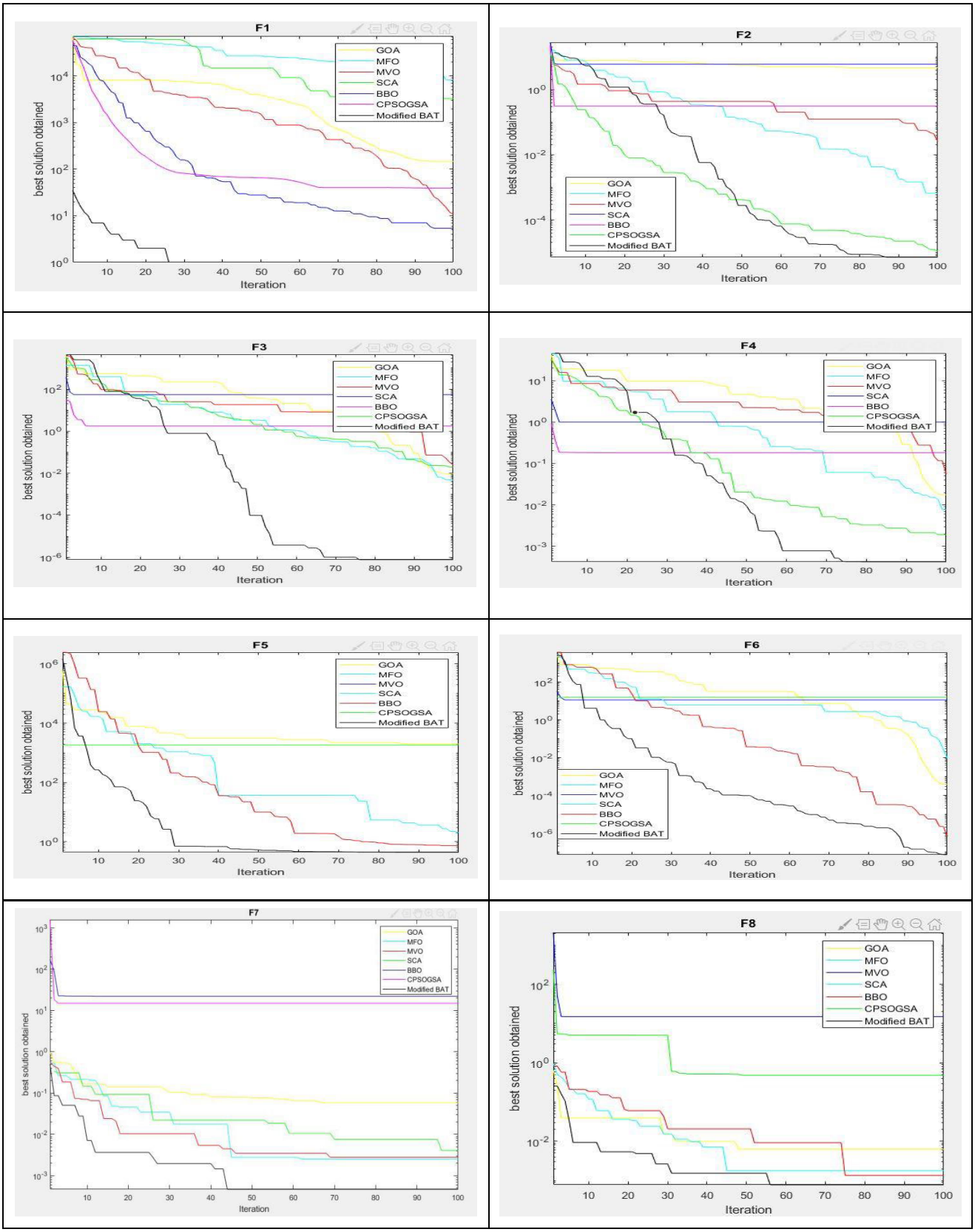

FIGURE 5. Comparison of convergence curve for f_1 to f_{14} test functions.

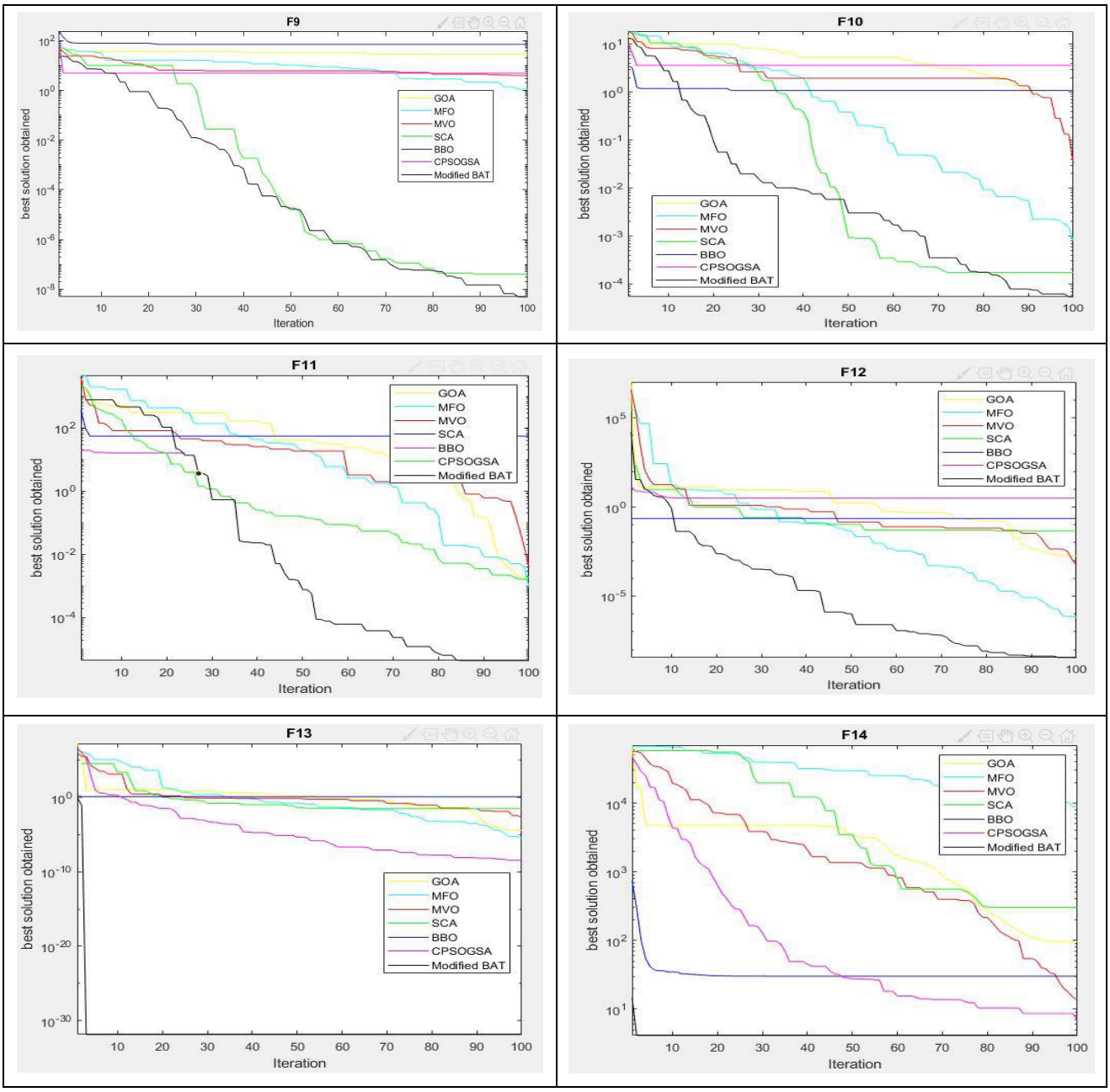

FIGURE 5. *(Continued.)* **Comparison of convergence curve for f_1 to f_{14} test functions.**

To validate the modified BAT algorithm performance, this section compares it to the performance of GOA, MFO, MVO, CPSOGA, BBO, and SCA. Figure 6 depicts the simulation results of the f1 to f14 test functions.

Figure 4 is the pictorial Explanation of 3D model of 14 various test function which adapted in this research work. Figure 6 is concern about Comparison of Convergence curve for f1 to f14 test functions. Correlation test that we've done both in this segment and in the Examination Test for Improper Integrals, that it won't generally be the denominator that is driving the assembly or disparity. Some of the time the numerator will decide whether something will merge or wander so don't get excessively locked. The examination tests are utilized to decide union or disparity of series with positive terms. If each term in one series is not exactly the relating term in some concurrent series, it should unite as well. In request for a series to join the series terms should go to focus in the breaking point. On the off chance that the series terms don't go to focus in the cutoff then it is absolutely impossible that the series can meet since this would violate the hypothesis Wilcoxon's rank-sum test [X] determines whether the Modified BAT results were statistically

TABLE 6. Statistical measures of various algorithms for f_1 to f_{14} benchmark functions.

Functions	Statistics	GOA	MFO	MVO	CPSOGA	BBO	SCA	**Modified BAT**	p value-Modified BAT
f_1	Std Dev	1.60E-004	7.14E+003	3.46E-001	3.51E-005	3.57E+003	8.65E-002	1.10E-004	NA
	Mean	1.96E-004	4.51E+003	1.43E+000	5.24E-005	2.25E+003	3.58E-001	1.09E-004	
	Best	1.16E-004	2.23E+000	6.99E-001	9.56E-006	1.11E+000	1.75E-001	1.03E-004	
f_2	Std Dev	3.30E-002	2.93E+001	5.23E+001	1.65E-002	1.46E+001	1.31E+001	1.00E-004	NA
	Mean	3.25E-002	4.41E+001	2.89E+001	1.62E-002	2.21E+001	7.24E+000	1.00E-004	
	Best	2.10E-003	6.59E-001	7.04E-001	1.00E-003	3.29E-001	1.76E-001	1.00E-004	
f_3	Std Dev	5.32E+001	1.63E+004	1.33E+002	2.66E+001	8.13E+003	4.65E+001	1.48E-004	NA
	Mean	1.77E+002	3.30E+004	3.75E+002	8.87E+001	1.65E+004	1.38E+002	1.64E-004	
	Best	1.02E+002	9.82E+003	1.53E+002	5.08E+001	4.91E+003	6.36E+001	1.14E-004	
f_4	Std Dev	2.27E-001	6.65E+000	1.34E+000	1.14E-001	3.32E+000	3.87E-001	5.69E-004	NA
	Mean	1.51E+000	6.85E+001	2.85E+000	7.50E-001	3.42E+001	1.09E+000	1.60E-003	
	Best	9.60E-001	5.16E+001	8.70E-001	4.81E-001	2.58E+001	4.58E-001	9.20E-004	
f_5	Std Dev	7.42E+001	1.27E+007	7.72E+002	3.71E+001	6.33E+006	2.12E+002	4.07E-002	NA
	Mean	1.15E+002	2.03E+006	6.22E+002	7.70E+001	1.01E+006	2.04E+002	3.89E+001	
	Best	3.45E+001	5.50E+002	4.83E+001	3.66E+001	2.94E+002	4.01E+001	3.88E+001	
f_6	Std Dev	1.98E-004	7.14E+003	3.08E-001	7.33E-002	3.57E+003	1.50E-001	1.47E-001	NA
	Mean	2.03E-004	4.51E+003	1.37E+000	4.32E+000	2.26E+003	4.67E+000	8.65E+000	
	Best	1.07E-004	3.41E+000	9.23E-001	4.15E+000	5.85E+000	4.37E+000	8.29E+000	
f_7	Std Dev	8.30E-002	8.46E+000	1.17E-002	4.15E-002	4.23E+000	2.37E-002	2.02E-004	NA
	Mean	2.38E-001	4.47E+000	3.68E-002	1.19E-001	2.24E+000	6.88E-002	2.13E-004	
	Best	7.16E-002	1.01E-001	1.71E-002	3.58E-002	5.01E-002	2.21E-002	1.01E-004	
f_8	Std Dev	1.30E+003	1.16E+003	7.93E+002	3.40E+003	3.33E+003	3.27E+003	5.50E+003	0.5489
	Mean	8.64E+003	1.12E+004	1.03E+004	1.18E+004	1.31E+004	1.22E+004	1.49E+004	
	Best	4.73E+003	1.36E+004	1.20E+004	1.12E+004	1.56E+004	1.30E+004	1.76E+004	
f_9	Std Dev	1.45E+001	4.24E+001	3.52E+001	7.26E+000	2.12E+001	1.24E+001	1.00E-004	NA
	Mean	7.59E+001	2.17E+002	1.72E+002	3.80E+001	1.09E+002	6.20E+001	1.00E-004	
	Best	5.08E+001	1.18E+002	1.07E+002	2.54E+001	5.88E+001	3.95E+001	1.00E-004	
f_{10}	Std Dev	2.95E-001	5.89E+000	5.81E-001	1.47E-001	2.95E+000	2.19E-001	1.00E-004	NA
	Mean	9.42E-002	1.69E+001	1.75E+000	4.71E-002	8.47E+000	4.61E-001	1.00E-004	
	Best	2.50E-003	1.10E+000	7.56E-001	1.20E-003	5.51E-001	1.90E-001	1.00E-004	
f_{11}	Std Dev	8.10E-003	6.75E+001	6.69E-002	6.80E-003	3.37E+001	2.15E-002	5.70E-003	NA
	Mean	6.10E-003	4.39E+001	8.15E-001	1.16E-002	2.20E+001	2.14E-001	1.73E-002	
	Best	1.00E-004	1.01E+000	6.18E-001	2.35E-007	5.03E-001	1.54E-001	1.00E-004	

TABLE 6. *(Continued.)* Statistical measures of various algorithms for f_1 to f_{14} benchmark functions.

f_{12}	Std Dev	1.24E-002	6.83E+007	1.32E+000	7.95E-003	3.41E+007	3.35E-001	3.70E-003	0.0578
	Mean	2.00E-003	1.92E+007	2.36E+000	1.35E-003	9.60E+006	5.91E-001	9.00E-004	
	Best	1.00E-004	3.77E+000	3.52E-002	8.92E-008	1.88E+000	8.78E-003	1.00E-004	
f_{13}	Std Dev	8.30E-003	6.48E+007	8.99E-002	2.50E-002	3.24E+007	4.54E-002	4.18E-002	NA
	Mean	5.10E-003	1.03E+007	1.81E-001	4.86E-002	5.13E+006	9.25E-002	9.22E-002	
	Best	1.03E-004	1.73E+001	9.54E-002	1.92E-002	8.65E+000	4.30E-002	3.85E-002	
f_{14}	Std Dev	9.11E-001	1.28E+000	1.00E-004	4.55E-001	6.41E-001	2.28E-001	1.00E-004	0.0102
	Mean	1.64E+000	1.49E+000	9.97E-001	1.32E+000	1.25E+000	1.16E+000	9.97E-001	
	Best	9.95E-001	9.95E-001	9.95E-001	9.15E-001	9.15E-001	9.15E-001	9.97E-001	

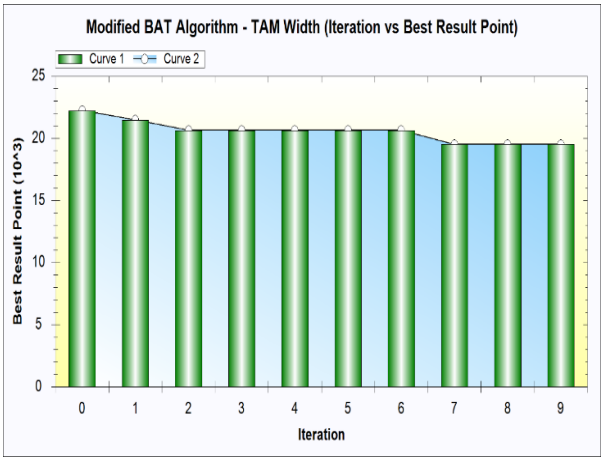

FIGURE 6. Iteration graph of D695 SoC using modified BAT algorithm.

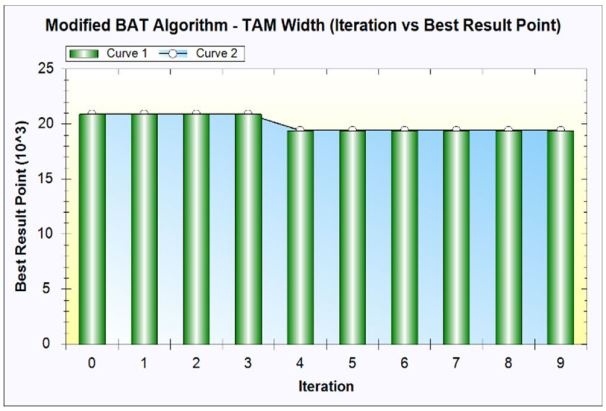

FIGURE 7. Iteration graph of P22810 SoC using modified BAT algorithm.

significant (5% significance level). Table 6 shows the p values for Modified BAT across all benchmark functions f_1 through f_{14}. In this table 6, NA stands for "not applicable," indicating that the related method could not be statistically compared in this test.

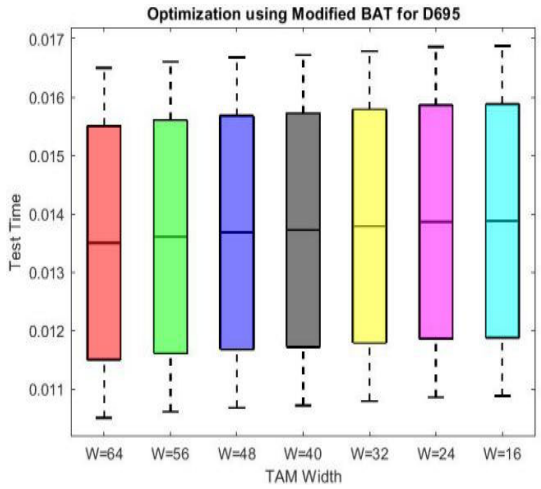

FIGURE 8. Box plot of test time and TAM width of modified BAT for D695 SoC benchmark.

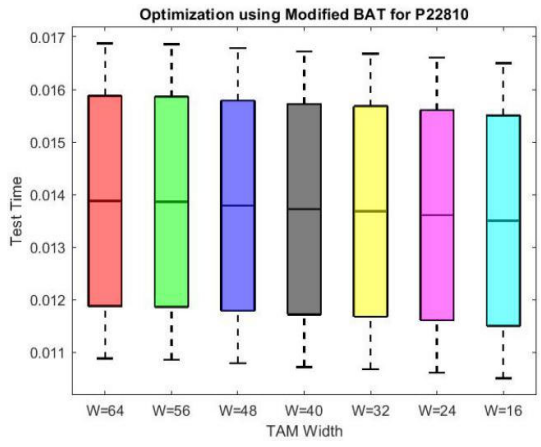

FIGURE 9. Box plot of test time and TAM width of modified BAT for P22810 SoC benchmark.

The algorithms convergence curves on f_1 to f_{14} functions are depicted in Figure 5. The Modified BAT method has a faster convergence rate, as shown in the diagram. Based on

TABLE 7. Test time of different algorithms for D695 SoC benchmark.

Algorithm	W= 64	W= 56	W= 48	W= 40	W= 32	W= 24	W= 16
GOA	0.0138928	0.0139421	0.0139870	0.0140606	0.0140916	0.0141273	0.0141813
MFO	0.0131595	0.0132153	0.0132578	0.0133275	0.0133570	0.0133908	0.0134420
MVO	0.0129175	0.0129730	0.0130147	0.0130832	0.0131121	0.0131453	0.0131955
CPSOGA	0.0126796	0.0127307	0.0127717	0.0128389	0.0128672	0.0129018	0.0129491
BBO	0.0132892	0.0133364	0.0133793	0.0134497	0.0134794	0.0135135	0.0135652
SCA	0.0136415	0.0137008	0.0137439	0.0138162	0.0138467	0.0138818	0.0139349
Modified BAT	0.0120726	0.0121143	0.0121534	0.0122171	0.0122442	0.0122754	0.0123223

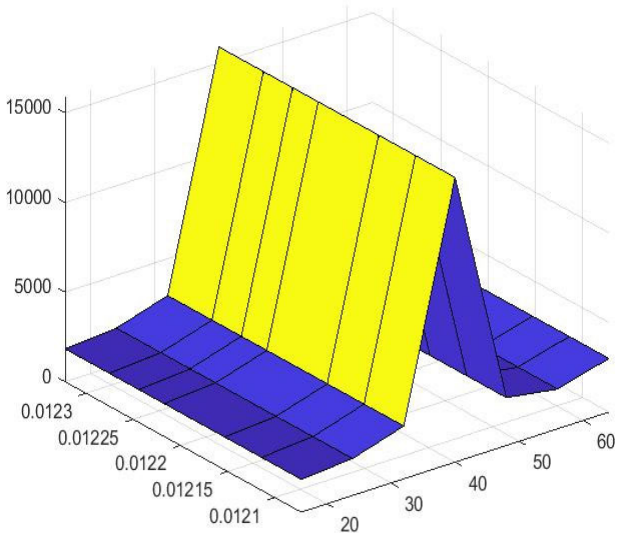

FIGURE 10. 3D plot of test time, TAM width of modified BAT - D695 SoC benchmark.

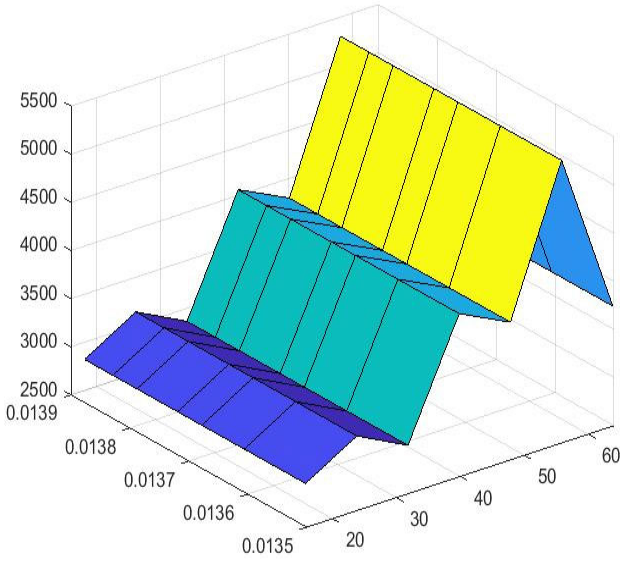

FIGURE 11. 3D plot of test time, TAM width of modified BAT –P22810 SoC benchmark.

these findings, it is possible to conclude that the Modified BAT technique can quickly find a global solution when dealing with benchmark functions.

The results show that the Modified BAT algorithm outperforms other optimization algorithms in terms of function optimization. The majority of the results show that the Modified BAT algorithm improves convergence speed and accuracy over other algorithms based on the testing functions of the convergence curves.

According to all test results, the improved BAT method improves convergence speed in most cases and shows a good optimization effect in convergence accuracy. When the algorithm's convergence speed and optimization correctness are guaranteed, the algorithm's robustness is enhanced.

Figures 6 and 7 depict Iteration and the Best Result Point for the D695 and P22810 SoCs when using the Modified BAT Algorithm.

Tables 8 and 9 show the test time values using the algorithms GOA, MFO, MVO, CPSOGA, BBO, SCA, and Modified BAT for the different TAM widths 64, 56, 48, 40, 32, 24, 16 of D695 and P22810 SoCs.

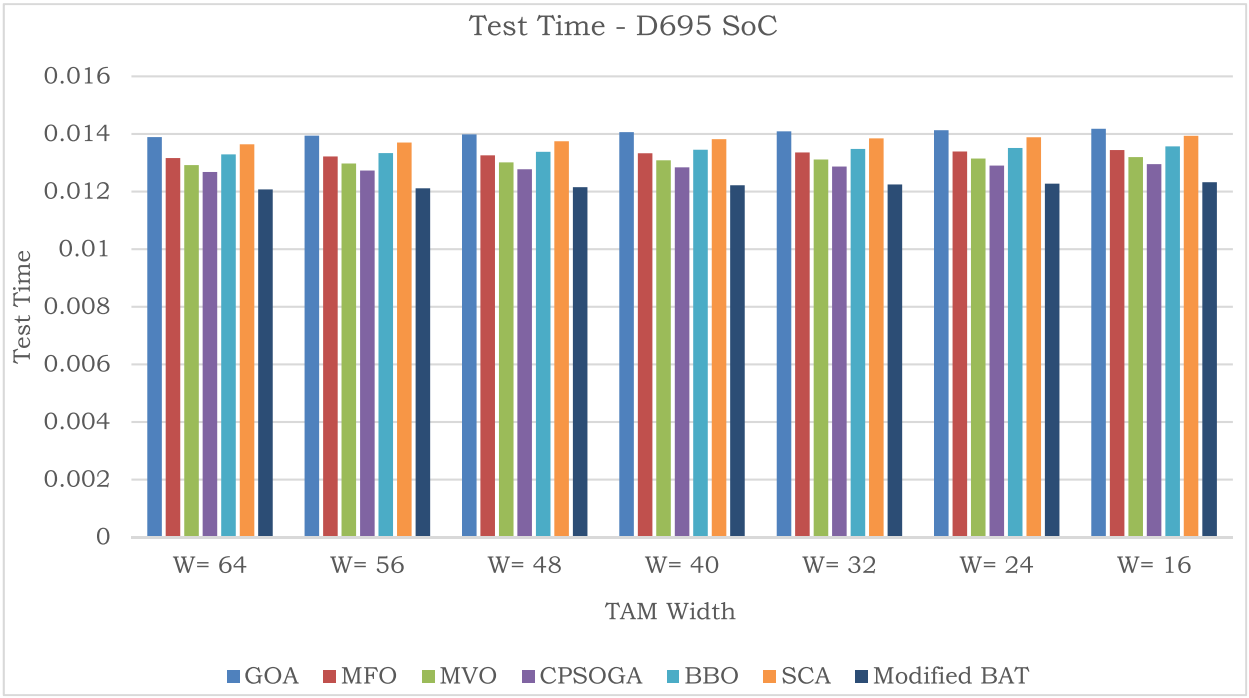

FIGURE 12. Graphical representation of test time for D695 SoC benchmark Circuit.

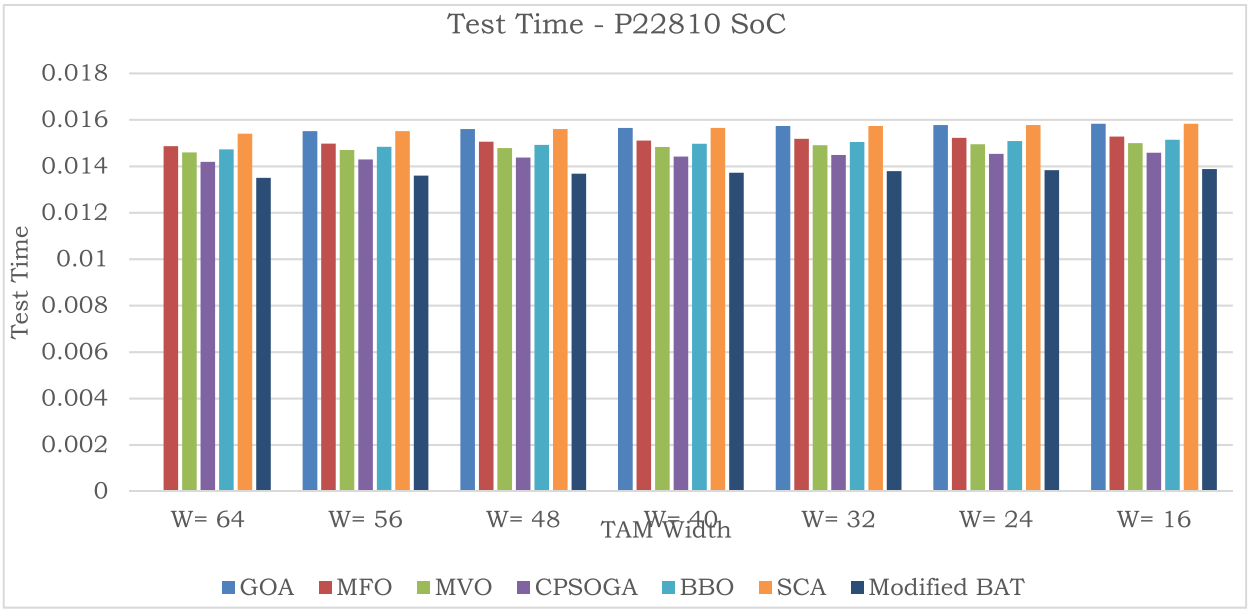

FIGURE 13. Graphical representation of test time for D695 SoC benchmark Circuit.

Figures 8 and 9 show the box plot of test time and TAM width of Modified BAT for D695 and P22810. In this container plot, The case length gives a sign of the example fluctuation and the line across the crate shows where the example is focused. The place of the case in its bristles and the place of the line in the container additionally lets us know whether the example is symmetric or slanted, either to one side or left long tell n box implies that values are slanted towards one side of the information or the other. Presently information can be emphatically or adversely slanted in the event that it's not balanced. In information that is emphatically slanted, the long tail is in the positive course. Figures 10 and 11 show a three-dimensional plot of Modified BAT test time and TAM width for D695 and P22810. This part presents an illustration of ongoing revolution of a 3D surface plot. The information utilized are from the Tests dataset. it is useful to see

TABLE 8. Test time of different algorithms for P22810 SoC benchmark.

Algorithm	W= 64	W= 56	W= 48	W= 40	W= 32	W= 24	W= 16
GOA	0.0154043	0.0155160	0.0156049	0.0156517	0.0157303	0.0157771	0.0158318
MFO	0.0148642	0.0149720	0.0150578	0.0151029	0.0151788	0.0152239	0.0152767
MVO	0.0145942	0.0147000	0.0147842	0.0148285	0.0149030	0.0149473	0.0149992
CPSOGA	0.0141891	0.0142920	0.0143739	0.0144170	0.0144894	0.0145325	0.0145829
BBO	0.0147292	0.0148360	0.0149210	0.0149657	0.0150409	0.0150856	0.0151379
SCA	0.0154043	0.0155160	0.0156049	0.0156517	0.0157303	0.0157771	0.0158318
Modified BAT	0.0135020	0.0136000	0.0136780	0.0137190	0.0137880	0.0138290	0.0138770

both the wires of the model and the 3D radiation plot at the equivalent time.plot3(X, Y, Z) plots arranges in three dimensional space. To plot a bunch of directions associated by line fragments, determine X, Y, and Z as vectors of a similar length. To plot numerous arrangements of directions on similar arrangement of tomahawks, determine something like one of X, Y, or Z as a network and the others as vectors. Three-layered (3D) impacts can be utilized to give profundity and add visual effect on diagrams in paginated reports.3D Representation is a significant piece of the plan cycle. It makes the plan cycle simpler for the architects and the clients. It permits the client to see a visual presentation of their home, even before the end result is done. This makes the plan cycle quicker and more straightforward. Figures 12 and 13 provide the test time for the different proposed algorithms for the two benchmarks.

The results show that the Modified BAT algorithm reduces testing time by a certain percentage when compared to the other algorithm. In comparison to GOA, MFO, MVO, CPSOGA, BBO, and SCA algorithms, the Modified BAT optimization algorithm reduces testing time by 15%, 9%, 7%, 5%, 10%, and 13% for d695 SoC and 14%, 8%, 7%, 5%, 9%, and 12% for p22810 SoC. As the TAM width increases, the testing time decreases due to the increased number of partitions.

VI. CONCLUSION

This paper solved the TAM optimization and test scheduling problems using a meta-heuristic approach. The proposed optimization algorithm is tested in SOCs D695 and P22810. Testing time of GOA, MFO, MVO, CPSOGA, BBO, SCA, and Modified BAT algorithms are estimated. Among the algorithms, the modified BAT algorithm performs better than the others. When compared with GOA, MFO, MVO, CPSOGA, BBO, and SCA algorithms, the test time of the Modified BAT Algorithm has been reduced to 15%, 9%, 7%, 5%, 10%, and 13% for d695 SoC and 14%, 8%, 7%, 5%, 9% and 12% for p22810. The proposed method outperforms the competition in terms of exploiting the optimum and provides advantages in terms of exploration, according to the findings. More research is required to develop an optimal list schedule and find the optimal number of partitions and partition locations. In the ITC'02 circuits, the Modified BAT algorithm outperformed other algorithms in finding optimal results in most cases. Algorithms such as Hybrid WOA-SA, Artificial Fish Swarm Algorithm, and Dragonfly Algorithm may be used in the future to reduce test time.

REFERENCES

[1] E. Larsson and Z. Peng, "An integrated system-on-chip test framework," in *Design, Automation, and Test in Europe*. 2008, pp. 439–454.
[2] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Test access mechanism optimization, test scheduling, and tester data volume reduction for system-on-chip," *IEEE Trans. Comput.*, vol. 52, no. 12, pp. 1619–1632, Dec. 2003.
[3] S. Goel, E. J. Marinissen, A. Sehgal, and K. Chakrabarty, "Testing of SoCs with hierarchical cores: Common fallacies, test access optimization, and test scheduling," *IEEE Trans. Comput.*, vol. 58, no. 3, pp. 409–423, Mar. 2009.
[4] E. J. Marinissen, V. Iyengar, and K. Chakrabarty, "A set of benchmarks for modular testing of SOCs," in *Proc. IEEE Int. Test Conf.*, Oct. 2002, pp. 519–528.

[5] K. Holzinger, V. Palade, R. Rabadan, and A. Holzinger, "Darwin or Lamarck? Future challenges in evolutionary algorithms for knowledge discovery and data mining," in *Interactive Knowledge Discovery and Data Mining in Biomedical Informatics*. 2014, pp. 35–56.

[6] C. Giri, S. Sarkar, and S. Chattopadhyay, "A genetic algorithm based heuristic technique for power constrained test scheduling in core-based SOCs," in *Proc. IFIP Int. Conf. Very Large Scale Integr.*, Oct. 2007, pp. 320–323.

[7] C. Xu, H. Hu, and J. Niu, "Test scheduling of SOC with power constraint based on particle swarm optimization algorithm," in *Proc. 3rd IEEE Int. Conf. Genet. Evol. Comput.*, Oct. 2009, pp. 611–614.

[8] W. Zou, S. M. Reddy, I. Pomeranz, and Y. Huang, "SOC test scheduling using simulated annealing," in *Proc. IEEE 21st VLSI Test Symp.*, May 2003, pp. 325–330.

[9] M. Dorigo and T. Stützle, "Ant colony optimization: Overview and recent advances," in *Handbook of Metaheuristics*. 2019, pp. 311–351.

[10] S. Banerjee, I. Mukherjee, and P. K. Mahanti, "Cloud computing initiative using modified ant colony framework," *World Acad. Sci., Eng. Technol.*, vol. 56, no. 32, pp. 221–224, 2009.

[11] S. Z. Mirjalili, S. Mirjalili, S. Saremi, H. Faris, and I. Aljarah, "Grasshopper optimization algorithm for multi-objective optimization problems," *Appl. Intell.*, vol. 48, no. 4, pp. 805–820, 2018.

[12] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowl.-Based Syst.*, vol. 89, pp. 228–249, Nov. 2015.

[13] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou, "Multi-verse optimizer: A nature-inspired algorithm for global optimization," *Neural Comput. Appl.*, vol. 27, no. 2, pp. 495–513, Feb. 2015.

[14] S. A. Rather and P. S. Bala, "A hybrid constriction coefficient-based particle swarm optimization and gravitational search algorithm for training multi-layer perceptron," *Int. J. Intell. Comput. Cybern.*, vol. 13, no. 2, pp. 129–165, Jun. 2020.

[15] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.

[16] S. A. Rather and P. S. Bala, "Analysis of gravitation-based optimization algorithms for clustering and classification," in *Handbook of Research on Big Data Clustering and Machine Learning*. Hershey, PA, USA: IGI Global, 2020, pp. 74–99.

[17] S. Mirjalili, "SCA: A sine cosine algorithm for solving optimization problems," *Knowl.-Based Syst.*, vol. 96, pp. 120–133, Mar. 2016.

[18] H. Wang, K. Li, and W. Pedrycz, "An elite hybrid Metaheuristic optimization algorithm for maximizing wireless sensor networks lifetime with a sink node," *IEEE Sensors J.*, vol. 20, no. 10, pp. 5634–5649, May 2020.

[19] C. Iwendi, P. K. R. Maddikunta, T. R. Gadekallu, K. Lakshmanna, A. K. Bashir, and J. Piran, "A metaheuristic optimization approach for energy efficiency in the IoT networks," *Softw., Pract. Exp.*, vol. 51, no. 12, pp. 2558–2571, 2021.

[20] M. Masdari and S. Barshandeh, "Discrete teaching–learning-based optimization algorithm for clustering in wireless sensor networks," *J. Ambient Intell. Hum. Comput.*, vol. 11, no. 11, pp. 5459–5476, Nov. 2020.

[21] T. A. Alghamdi, "Parametric analysis on optimized energy-efficient protocol in wireless sensor network," *Soft Comput.*, vol. 25, no. 6, pp. 4409–4421, Mar. 2021.

[22] A. Mishra, S. Choudhary, M. Vats, and S. Sachan, "LEACH with pheromone energy efficient routing in wireless sensor network," in *Intelligent Computing in Engineering*. Singapore: Springer, 2020, pp. 91–98.

[23] B. P. Dahiya, S. Rani, and P. Singh, "Lifetime improvement in wireless sensor networks using hybrid grasshopper meta-heuristic," in *Proceedings of ICRIC 2019*. Cham, Switzerland: Springer, 2020, pp. 305–320.

[24] A. Anurag, R. Priyadarshi, A. Goel, and B. Gupta, "2-D coverage optimization in WSN using a novel variant of particle swarm optimisation," in *Proc. 7th Int. Conf. Signal Process. Integr. Netw. (SPIN)*, Feb. 2020, pp. 663–668.

[25] A. B. S. Yıldız, N. Pholdee, S. Bureerat, A. R. Yıldız, and S. M. Sait, "Sine-cosine optimization algorithm for the conceptual design of automobile components," *Mater. Test.*, vol. 62, no. 7, pp. 744–748, Jul. 2020.

[26] N. Panagant, N. Pholdee, S. Bureerat, A. R. Yıldız, and S. M. Sait, "Seagull optimization algorithm for solving real-world design optimization problems," *Mater. Test.*, vol. 62, no. 6, pp. 640–644, 2020.

[27] B. S. Yıldız, "The mine blast algorithm for the structural optimization of electrical vehicle components," *Mater. Test.*, vol. 62, no. 5, pp. 497–502, May 2020.

[28] A. R. Yıldız, H. Özkaya, M. Yıldız, S. Bureerat, B. S. Yıldız, and S. M. Sait, "The equilibrium optimization algorithm and the response surface-based metamodel for optimal structural design of vehicle components," *Mater. Test.*, vol. 62, no. 5, pp. 492–496, May 2020.

[29] V. Patel, B. Raja, V. Savsani, and A. R. Yıldız, "Qualitative and quantitative performance comparison of recent optimization algorithms for economic optimization of the heat exchangers," *Arch. Comput. Methods Eng.*, vol. 28, no. 4, pp. 2881–2896, Jun. 2021.

[30] M. M. Fouad, A. I. Hafez, and A. E. Hassanien, "Optimizing topologies in wireless sensor networks: A comparative analysis between the grey wolves and the chicken swarm optimization algorithms," *Comput. Netw.*, vol. 163, Nov. 2019, Art. no. 106882.

[31] R. Sarangkum, K. Wansasueb, N. Panagant, N. Pholdee, S. Bureerat, A. R. Yıldız, and S. M. Sait, "Automated design of aircraft fuselage stiffeners using multiobjective evolutionary optimisation," *Int. J. Vehicle Des.*, vol. 80, nos. 2–4, pp. 162–175, 2019.

[32] J. Li, W. Tang, J. Wang, and X. Zhang, "A multilevel color image thresholding scheme based on minimum cross entropy and alternating direction method of multipliers," *Optik*, vol. 183, pp. 30–37, Apr. 2019.

[33] L. Xu, H. Jia, C. Lang, X. Peng, and K. Sun, "A novel method for multilevel color image segmentation based on dragonfly algorithm and differential evolution," *IEEE Access*, vol. 7, pp. 19502–19538, 2019.

[34] P. Kandhway and A. K. Bhandari, "A water cycle algorithm-based multilevel thresholding system for color image segmentation using Masi entropy," *Circuits, Syst., Signal Process.*, vol. 38, no. 7, pp. 3058–3106, 2019.

[35] K. P. B. Resma and M. S. Nair, "Multilevel thresholding for image segmentation using krill herd optimization algorithm," *J. King Saud Univ., Comput. Inf. Sci.*, vol. 33, no. 5, pp. 528–541, Jun. 2021.

[36] B. S. Yıldız and A. R. Yıldız, "Comparison of grey wolf, whale, water cycle, ant lion and sine-cosine algorithms for the optimization of a vehicle engine connecting rod," *Mater. Testing*, vol. 60, no. 3, pp. 311–315, 2018.

[37] V. Sheshadri, V. D. Agrawal, and P. Agrawal, "Power-aware SoC test optimization through dynamic voltage and frequency scaling," in *Proc. IFIP/IEEE 21st Int. Conf. Very Large Scale Integr. (VLSI-SoC)*, Oct. 2013, p. 2013.

[38] J. Shao, G. Ma, Z. Yang, and R. Zhang, "Process algebra based SoC test scheduling for test time minimization," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*, Apr. 2008, pp. 134–138.

[39] G. Rohini and S. Salivahanan, "Optimal test time for system-on-chip designs using fuzzy logic and process algebra," *J. Comput. Sci.*, vol. 6, no. 1, pp. 12–17, Jan. 2010.

[40] W. Marrouche, R. Farah, and H. M. Harmanani, "A strength Pareto evolutionary algorithm for optimizing system-on-chip test schedules," *Int. J. Comput. Intell. Appl.*, vol. 17, no. 2, Jun. 2018, Art. no. 1850010.

[41] D. Karaboga and B. Akay, "A comparative study of artificial bee colony algorithm," *Appl. Math. Comput.*, vol. 214, no. 1, pp. 108–132, Aug. 2009.

[42] G. Chandrasekaran, S. Periyasamy, and P. R. Karthikeyan, "Test scheduling for system on chip using modified firefly and modified ABC algorithms," *Social Netw. Appl. Sci.*, vol. 1, no. 9, p. 1079, Sep. 2019.

[43] X.-S. Yang and X. He, "Firefly algorithm: Recent advances and applications," *Int. J. Swarm Intell.*, vol. 1, no. 1, pp. 36–50, 2013.

[44] G.-H. Xu, T.-W. Zhang, and Q. Lai, "A new firefly algorithm with mean condition partial attraction," *Appl. Intell.*, vol. 52, pp. 4418–4431, 2021.

[45] X. Yang and X. He, "Bat algorithm: Literature review and applications," *Int. J. Bio-Inspired Comput.*, vol. 5, no. 3, pp. 141–149, Aug. 2013.

[46] X. Yang and A. H. Gandomi, "Bat algorithm: A novel approach for global engineering optimization," *Eng. Comput.*, vol. 29, no. 5, pp. 464–483, Jul. 2012.

[47] S. Yılmaz, E. U. Kucuksille, and Y. Cengiz, "Modified bat algorithm," *Electron. Electr. Eng.*, vol. 20, no. 2, pp. 71–78, Feb. 2014.

[48] S. Mirjalili, S. M. Mirjalili, and X.-S. Yang, "Binary bat algorithm," *Neural Comput. Appl.*, vol. 25, nos. 3–4, pp. 663–681, Sep. 2014.

[49] X. Cai, X.-Z. Gao, and Y. Xue, "Improved bat algorithm with optimal forage strategy and random disturbance strategy," *Int. J. Bio-Inspired Comput.*, vol. 8, no. 4, pp. 205–214, 2016.

[50] A. Alihodzic and M. Tuba, "Improved bat algorithm applied to multilevel image thresholding," *Sci. World J.*, vol. 2014, pp. 1–16, Aug. 2014.

[51] R. Qamar, N. Bajao, I. Suwarno, and F. A. Jokhio, "Survey on generative adversarial behavior in artificial neural tasks," *Iraqi J. Comput. Sci. Math.*, vol. 3, no. 2, pp. 83–94, 2022.

[52] K. Aggarwal, M. M. Mijwil, A. H. Al-Mistarehi, S. Alomari, M. Gök, A. M. Z. Alaabdin, and S. H. Abdulrhman, "Has the future started? The current growth of artificial intelligence, machine learning, and deep learning," *Iraqi J. Comput. Sci. Math.*, vol. 3, no. 1, pp. 115–123, 2022.

GOKUL CHANDRASEKARAN received the B.E. degree in electrical and electronics engineering from Anna University, India, in 2008, the M.E. degree in VLSI design from the Government College of Technology Coimbatore, India, in 2010, and the Ph.D. degree from the Electronics and Communication Engineering Department, Anna University. His research interests include VLSI design and testing, electric vehicle machine learning, image processing, and computer vision.

NEELAM SANJEEV KUMAR received the Ph.D. degree in evolutionary computing technique from the Department of Electronics and Communication Engineering, Anna University, Chennai, India. He is currently associated with the Saveetha School of Engineering as an Assistant Professor. His research interests include the Internet of Things, privacy and security in networks, big data, cloud computing, data mining and analysis, prediction, optimization, and machine learning.

KARTHIKEYAN P. R. received the B.E. degree in electrical engineering from Anna University, India, in 2008, the M.Tech. degree in embedded system technology from SRM University, India, in 2010, and the Ph.D. degree from the Electronics and Communication Engineering Department, Anna University, in 2019. His research interests include machine learning, image processing, and computer vision.

VANCHINATHAN K. received the Bachelor of Engineering degree in electrical and electronics engineering, in 2007, the Master of Engineering degree in power electronics and drives, in 2009, and the Ph.D. degree from the Faculty of Electrical Engineering, Anna University. He has eight years of experience in teaching, industry, and research field.

NEERAJ PRIYADARSHI (Senior Member, IEEE) received the M.Tech. degree in power electronics and drives from the Vellore Institute of Technology (VIT), Vellore, India, in 2010, and the Ph.D. degree from the Government College of Technology and Engineering, Udaipur, Rajasthan, India.

He is currently associated with the Department of Electrical Engineering, JIS College of Engineering, Kolkata, India. He is also associated with Bioenergy and Green Engineering, Department of Energy Technology, Aalborg University. He has a Post Doc in bioenergy and green engineering at the Department of Energy Technology, Aalborg University. In previous postings he was with BITT Ranchi, MIT Purnea, JK University, Geetanjali Institute, Global Institute, and SS Group, Rajasthan, India. He has published over 60 papers in journals and conferences of high repute such as, IEEE Systems Journal, *IET Electric Power Applications*, *IET Power Electronics*, IEEE Access Journal, *Electric Power Components and Systems*, *International Transaction of Electrical Energy Systems*, Wiley *Energies* MDPI, and *International Journal of Renewable Energy Research*. His current research interests include power electronics, control systems, power quality, and solar power generation.

Dr. Priyadarshi successfully organized two international workshops and one National workshop in different organization. He is also invited as a Guest speaker in many international/national conferences. He is a Reviewer of IEEE Systems Journal, *Electric Power Components and Systems*, Taylor and Francis, IEEE Access, *IET Renewable Power Generation*, *International Journal of Modeling and Simulation*, *International Journal of Renewable Energy Research*, *International Journal of Power Electronics and Drives*, and reputed Elsevier SCI indexed journals. Under him guidance 25 DST, Government of Rajasthan projects have been sanctioned for financial support. He is a Chief Editor of four edited books of renowned publishers (*Intelligent Renewable Energy Systems: Integrating Artificial Intelligence Techniques and Optimization Algorithms* (Wiley) and *Advances in Power Systems and Energy Management* (Springer), and De Gruyter Publisher, Germany).

BHEKISIPHO TWALA (Senior Member, IEEE) is working as a Research Scientist at Digital Transformation Portfolio, Tshwane University of Technology, Pretoria West, Pretoria, South Africa, with over 20 years experience in putting mathematics to scientific use in the form of data comparison, inference, analysis, and presentation to design, collect, and interpret data experiments surrounding the fields of transport, medical, artificial intelligence, software engineering, robotics and most recently in electrical and electronic engineering science. A track record of publications in conferences and journals and the ability to communicate scientific ideas to an informed lay audience. Plans for future research work will, of course, be influenced by the ideas and opportunities which present themselves at my employer, but the intention is to continue exploring problems in the areas of machine learning (artificial intelligence) and statistics and be committed to the highest levels of professional and personal excellence.